

MULTIAGENT NEGOTIATION FOR FAIR AND UNBIASED ALLOCATION OF  
HYPERDIMENSIONAL RESOURCES

by

Karthik Iyer

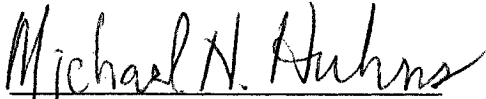
Bachelor of Engineering  
Mumbai University, 1997


Master of Technology  
IIT Bombay, 2000

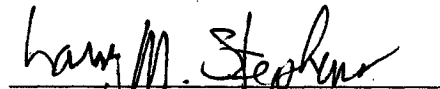
---

Submitted in Partial Fulfillment of the Requirements  
for the Degree of Doctor of Philosophy in the  
Department of Computer Science and Engineering  
College of Engineering and Computing  
University of South Carolina

2007

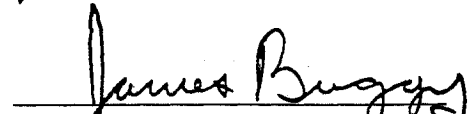
  
Major Professor

  
Chairman Examining Committee

  
Committee Member

  
Committee Member

  
Committee Member

  
Dean of the Graduate School

UMI Number: 3280325

Copyright 2007 by  
Iyer, Karthik

All rights reserved.

### INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

**UMI**<sup>®</sup>

---

UMI Microform 3280325

Copyright 2007 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company  
300 North Zeeb Road  
P.O. Box 1346  
Ann Arbor, MI 48106-1346

© Copyright by Karthik Iyer, 2007

All Rights Reserved.

## Acknowledgements

I am deeply indebted to my guru Mani Amma whose guiding light is the reason for my every successful endeavor in life. My mom, Kamakshi Rangarajan and dad, Mannargudi Subramanya Rangarajan who stood by me steadfastly through all my ups and down and believed in their son. My "Thambi" (younger brother), Kaushik Rangarajan whose kept my enthusiasm alive during harsh times. My advisor, mentor and guide, Dr. Michael Huhns, for giving me the opportunity to come to the US and pursue my PhD. I also thank him for being patient and understanding when i faced difficulties. My dissertation committee members, Dr. John Rose, Dr. Marco Valtorta, Dr. Larry Stephens and Dr. Stanley Dubinsky for their understanding support and guidance. My late grandparents, Nani and Thatha whose kindness and love i still miss. The faculty and staff of the University of South Carolina for the providing me an important educational opportunity and a home away from home. My friends who were my family here in Columbia and simply for having a whale of a time. Last but not least the people of Columbia for their warm and friendly nature and their eagerness to help out.

Thank You!

## Abstract

Negotiation in multiagent systems is a topic of active research in the area of distributed computing. Agents are autonomous software entities that represent their owner's interests. As software agents interact in the spheres of commerce and online trading, it becomes essential to focus on design of an appropriate interaction mechanism. Since there is no global control in a multiagent system, one can only induce agents to behave in desirable ways. Therefore, negotiation protocols should resist manipulation by agents, induce agents to reveal their real valuations, and allow for fair distribution of resources. Mathematicians have analyzed similar problems in the domain of cake cutting wherein resources have to be allocated among competing humans. This dissertation analyzes and applies ideas from cake cutting to the design and analysis of negotiation protocols. The problem is to come up with a negotiation protocol that enables allocation of resources among agents while satisfying various desirable properties. An extensive literature survey is included that covers relevant existing work in the fields of mathematics, economics, social justice, and multiagent systems itself. The dissertation goes beyond simply extending the ideas in the literature. As part of its original contribution to the field of multiagent negotiation, firstly, an extensive list of negotiation criteria is presented that has been culled from the literature. In addition, I propose three new criteria: verifiability, dimensionality, and topology. The intention is to analyze existing protocols exhaustively as well as to design new protocols from the ground up. Second, I present protocols and procedures for one-dimensional linear and circular resource allocation. Third, this work is extended to account for two-dimensional planar resources. Fourth, I propose protocols

and procedures for allocation of two-dimensional cylindrical resources. These two-dimensional procedures are analyzed in detail for their benefits and drawbacks. Finally, this work ends with a demonstration of how an important result in mathematical combinatorics called Sperner's lemma can be used to enable fair divisions. The various open problems that need to be solved before Sperner's lemma can be used for fair as well as efficient allocations are discussed in detail.

## Table of Contents

Acknowledgements.....	iii
Abstract.....	iv
List of Tables.....	xii
List of Figures.....	xiii
1. Introduction.....	1
1.1. Agents and Multiagent systems.....	1
1.2. Multiagent Negotiation.....	5
1.3. Fair Division Schemes.....	14
1.4. Motivations.....	18
2. Literature Survey.....	30
2.1. Generalized Multiagent Negotiation.....	30
2.2. Cake Cutting Protocols.....	38
2.2.1. One-Dimensional Resource Sharing.....	38
2.2.2. Hyperdimensional Resource Sharing.....	43
2.2.3. Fair Division Schemes For Cake Cutting.....	54
2.2.3.1. Background.....	54
2.2.3.2. Proportional Fair Division.....	55
2.2.3.3. Maxmin Fairness.....	72
2.2.3.4. Envy-Free Fair Division.....	74
2.2.3.5. Redefining Envy-Free.....	79
3. Evaluating Negotiation Protocols and Procedures.....	85
3.1. Protocol vs. Procedure.....	87

3.2.	Protocol Engineering .....	90
3.3.	Criteria for Allocations .....	91
3.3.1.	Fairness .....	92
3.3.2.	Envy-freeness.....	96
3.3.2.1.	Alternatives to Envy-Freeness .....	97
3.3.2.2.	Envy-Minimizing Criteria.....	98
3.3.2.3.	Envy-Free Criteria .....	101
3.3.3.	Exact .....	102
3.3.4.	Equitability.....	103
3.3.5.	Efficiency .....	104
3.3.6.	Relationship between Various Allocation Criteria .....	108
3.4.	Criteria for Protocols.....	109
3.4.1.	Symmetric [7] .....	110
3.4.2.	Simplicity [7] .....	111
3.4.3.	Stability [7] .....	111
3.4.4.	Anonymity and Privacy .....	112
3.4.5.	Strategy-Proof [7] .....	113
3.4.6.	Centralized vs. Distributed.....	113
3.4.7.	Mediated vs. Arbitrated.....	114
3.4.8.	Communication Complexity [87] .....	115
3.4.9.	Time [26] .....	116
3.4.10.	One-Shot vs. Iterative .....	117
3.5.	Criteria for Procedures.....	118



3.5.1.	Bounded vs. Unbounded Number of Agents .....	118
3.5.2.	Discrete vs. Continuous .....	119
3.5.3.	External Payments .....	121
3.5.4.	Verifiability [49] .....	122
3.5.5.	Computational Complexity .....	123
3.5.6.	Division Independence [50] .....	124
3.6.	Criteria for Resources .....	125
3.6.1.	Divisibility .....	126
3.6.2.	Homogeneous vs. Heterogeneous .....	127
3.6.3.	Sharable or Not [52] .....	128
3.6.4.	Measurability [47] .....	128
3.6.5.	Dimensionality .....	129
3.6.6.	Topology .....	130
3.6.7.	Resources vs. Tasks (Desirable vs. Undesirable) (Goods vs. Bads) [52] ...	132
3.6.8.	Single vs. Multiunit [52] .....	133
3.6.9.	Static vs. Perishable [52] .....	133
3.6.10.	Consumable vs. Resusable [52] .....	134
3.7.	Criteria for Utility Functions .....	134
3.7.1.	Non-Atomic .....	134
3.7.2.	Additive .....	135
3.7.3.	Concave .....	135
3.7.4.	Continuous .....	135
4.	Protocols and Procedures for One-dimensional Resource Allocation .....	137

4.1.	Dividing a Linear Resource among n Agents .....	138
4.1.1.	Protocol .....	138
4.1.2.	Features .....	144
4.2.	A Fair and Unbiased Protocol to Partition a Circular Resource among n Agents 145	
4.2.1.	Protocol .....	145
4.2.2.	Features .....	155
4.3.	Conclusion and Discussion .....	156
5.	Protocols and Procedures for Two-dimensional Planar Resource Allocation .....	159
5.1.	Motivation .....	161
5.1.1.	Cake Cutting .....	161
5.1.2.	Land Distribution .....	162
5.1.3.	Allocating Radio Spectrum .....	162
5.1.4.	Case for Higher Efficiency .....	164
5.2.	Dividing a Two-Dimensional Resource among n Agents .....	165
5.2.1.	Protocol .....	166
5.2.2.	Procedure given the non-overlap condition .....	168
5.2.3.	Features of Procedure for the non-overlap condition .....	170
5.2.4.	Degree of Partial Overlap .....	171
5.2.5.	Procedure given The overlap degree condition .....	173
5.2.6.	An Example Allocation Problem .....	177
5.2.7.	Proof given The overlap degree condition .....	181
5.2.8.	Features of Procedure for The overlap degree condition .....	184

5.3.	Conclusion .....	185
6.	Protocols and Procedures for Two-dimensional Cylindrical Resource Allocation	187
6.1.	Motivation.....	188
6.2.	Topology of a Cylindrical Surface.....	191
6.3.	Dividing a Two-Dimensional Cylindrical Resource among n Agents .....	196
6.3.1.	Mapping the Cylindrical RA Problem to the Circular RA Problem.....	198
6.3.2.	Mapping the Cylindrical RA Problem to the Linear RA Problem.....	201
6.3.3.	A Native Cylindrical RA Procedure .....	206
6.3.4.	An Example Allocation Problem.....	216
6.3.5.	Features of the Native Cylindrical RA Procedure .....	224
6.4.	Conclusion .....	225
7.	Issues in Two-dimensional Resource Allocation Procedures.....	227
7.1.	Discussion.....	227
7.2.	Future Work.....	240
8.	On the Applicability of Sperner's Lemma for Multiagent Resource Allocation....	245
8.1.	The Mechanics of Sperner's Lemma.....	246
8.1.1.	Triangulation.....	248
8.1.2.	Sperner Labeling.....	249
8.1.3.	Procedure for Locating a Fully Labeled Elementary Simplex.....	251
8.2.	Applying Sperner's Lemma to Multiagent Resource Allocation .....	253
8.2.1.	Conventions and Assumptions.....	254
8.2.2.	Approximate Envy-free Procedure for Multiagent Resource Allocation ...	255
8.2.3.	Approximate-fair Procedure for Multiagent Resource Allocation .....	260

8.3.	Discussion.....	265
8.3.1.	Improving the Efficiency of Allocation.....	265
8.3.2.	Devising Algorithms for Exhaustive Search.....	268
8.3.2.1.	Sub-problem: How Many Labeled States Exist for an n-Simplex?.....	270
8.3.2.2.	Sub-sub-problem: Given a number n, how many distinct sets of numbers can be created, such that the members in each set add up to n? .....	274
8.4.	Conclusion .....	285
9.	Conclusion .....	287
	References.....	291

## List of Tables

Table 1. The valuations shown are from the Perspective of Agent 1. The allocations show how approximate fair and approximate envy-free relate to each other.....	109
Table 2. Various Problem Domain Types.....	127
Table 3. Relationships between two L-intervals based on the ordering of the points ....	210
Table 4. Relationships between two C-intervals based on the ordering of the points ...	214
Table 5. The 12 topologically unique cases of overlaps along the C-axis.....	215
Table 6. Generating number sets using recursive rules. The last column shows the sets that have been missed by the rules.....	275
Table 7. Generating number sets by determining the combinations in which the coins can be put in slots. When the number of slots available is 2 or 3, the function $F(n,k)$ is unknown.....	277
Table 8. Values of $B(n)$ for $1 \leq n \leq 100$ generated by the java program. ....	283

## List of Figures

- Figure 1. Space of possible deals for agents  $i$  and  $j$ . This is an elaboration of the figure drawn by Woolridge [1]. The polygon is used to represent any closed convex shaped region. .... 9
- Figure 2. A political cartoon illustrating the electoral districts drawn by the Massachusetts legislature to favour the incumbent Democratic-Republican party candidates. The image is in public domain. .... 25
- Figure 3. A standard drawing of districts ( dashed lines). The squares and the triangles represent voters with different party affiliations. A total of four districts need to be drawn. Thus two districts will be won by the triangle party and two by the square party. .... 26
- Figure 4. A representation of how “packing” is done. Since the voting mechanism is nonproportional, all constituents of the triangle party can be packed into one district, while the remaining districts go to the square party. In this case squares get an advantage by winning three seats out of four” ..... 27
- Figure 5. A representation of how “cracking” is done. Here the triangle party redraws the districts in such a way that it breaks up square majority areas. With a nonproportional system, a simple majority is sufficient for a candidate to win. Due to this all four districts are won by the triangle party. .... 28
- Figure 6. Agent A’s demarcates of its share of the region. The thick line represents the land allocated to A ..... 46

Figure 7. Different ways of measuring envy-freeness .....	84
Figure 8. How strategies are constrained by rules: in this example only whole cent amounts are allowed in the negotiation. ....	88
Figure 9. Engineering a protocol. ....	94
Figure 10. How max-min fairness works.....	95
Figure 11. A general view of how discrete allocation procedures work .....	107
Figure 12. Relationships between the various equity criteria. ....	110
Figure 13. Cake cutting using different topologies for rectangular and circular shaped cakes. The arrows show the direction of knife movement. The figure shows example allocations for three agents. ....	131
Figure 14. Different resources can have different topologies. Procedures should be designed with these topologies in mind to generate more efficient solutions .....	131
Figure 15. The linear cake cutting protocol.....	139
Figure 16. The circular cake cutting protocol.....	147
Figure 17. Conventions used in the proof.....	150
Figure 18. Allocations for $n=2$ for a circular resource .....	152
Figure 19. Various flavors of icing on a cake .....	161
Figure 20. Land with assets. ....	162
Figure 21. Allocating frequency spectrum.....	163

Figure 22. a. Resources with negative utilities. b. Allocating using modified moving knife. ....	164
Figure 23. Agents $i, j$ , and $k$ mark portions that fulfill the non-overlap condition. The labels of the rectangles denote their respective owners. ....	167
Figure 24. Projections of the agents' portions onto the Y-axis.....	169
Figure 25. Transforming the two-dimensional problem into the one-dimensional case	169
Figure 26. The different types of overlap between two rectangles.....	172
Figure 27. The scoring matrix for a rectangle. The types of interval overlaps are Separate(S), Partial(P), Superset(Sp), Subset(Sb). ....	174
Figure 28. Agents $i, j$ , and $k$ mark portions that fulfill the overlap degree condition. The labels of the rectangles denote their respective owners. ....	179
Figure 29. The directed graph based on the topology of overlapping rectangles. ....	180
Figure 30. Bidding for frequency bands in the modified FCC auction .....	189
Figure 31. Bidding for seating blocks in a sports stadium.....	190
Figure 32. Transformation a two-dimensional rectangle to a cylindrical surface. ....	191
Figure 33. Generating cylindrical-shaped surfaces based on the various values of parameters.....	193
Figure 34. Drawing a rectangle on a cylindrical surface .....	195
Figure 35. Topologies of planar and cylindrical resources.....	195
Figure 36. The dimensions of a cylindrical resource.....	198



Figure 37. Projection of rectangles onto the circular dimension. ....	200
Figure 38. Agents $i$ and $j$ mark out two rectangles each on the cylindrical surface. Agent $i$ 's rectangles are represented by solid lines, while agent $j$ 's rectangles are marked by the dotted lines. ....	203
Figure 39. Projections of the agents' portions onto the L-axis.....	204
Figure 40. Transforming the two-dimensional cylindrical RA into the one-dimensional linear RA.....	205
Figure 41. Nature of overlap of agent rectangles along the C-axis and the L-axis.....	208
Figure 42. The topologically unique overlaps of L-intervals .....	210
Figure 43. Interpreting an interval: Based on the placement of the start mark and direction of rotation, it is possible to encounter the end mark ( $e_1$ ) of an interval before the start mark ( $s_1$ ). The sequence of points maps to case 16 in Table 4 and the deduced intervals $I_1$ and $I_2$ are shown.....	212
Figure 44. The different types of overlap between two rectangles on a cylindrical surface. Only 9 of the 36 possible combinations are shown.....	217
Figure 45. Agents $j$ and $k$ mark out portions on a cylindrical resource that fulfill the overlap degree condition. All projections along the C-axis are shown. Only one projection onto the L-axis is shown to reduce clutter. ....	218
Figure 46. The scoring matrix for a rectangle. The types of interval overlaps are Separate(S), Partial(P), Superset(Sp), Subset(Sb) .....	220
Figure 47. The directed graph based on the topology of overlapping rectangles.....	221

Figure 48. The non-overlap condition and the overlap degree condition are true. ....	228
Figure 49. The non-overlap condition is true and the overlap degree condition is false. .....	230
Figure 50. The non-overlap condition is false and the overlap degree condition is true. .....	230
Figure 51. The solution space for Iyer's conditions. ....	231
Figure 52. Agent 1 divides the land into two vertical strips, while agent 2 divides them into two horizontal strips. ....	232
Figure 53. Triangulation of a 2-simplex (triangle) into elementary simplices. The (1,2,3) represent Sperner labeling of the triangulation. ....	247
Figure 54. An example of a 2-simplex (triangle) embedded in 3-dimensional Euclidean space. ....	248
Figure 55. An arbitrary triangulation of a triangle. The vertices are labeled as per Sperner labeling rules. ....	250
Figure 56. Kuhn's procedure to locate a fully labeled elementary simplex. ....	253
Figure 57. Labeling of the triangulated 2-simplex with agent names. A, B and C denote the names of the agents. ....	256
Figure 58 part A. Flowchart of approximate envy-free / approximate-fair procedure to allocate resources. Figure continues in part B below. ....	258
Figure 59. An example showing how the label of a vertex is determined. $M_1$ , $M_2$ , and $M_3$ are the coordinates of the vertex at (0.22, 0.33, 0.45). $A_1$ , $A_2$ , and $A_3$ are the owner's	

marks respectively. In this example, the vertex would be labeled 3 since $M_3$ is a superset of $A_3$ .	262
Figure 60. A Sperner labeled triangulation of an 2-simplex. The shaded triangles are the fully labeled elementary simplices, $\sigma^2$ , and the 1-2 segments are $\sigma^1$ .	266
Figure 61. NFA for a 1-simplex (line). The desired final state is shown in bold and the $\lambda$ -transitions are shown by the dotted lines.	268
Figure 62. The NFA for a 2-simplex (triangle). The labeled states represent the types of labeled elementary simplices. The bold circle shows the FLES and is the “success” state. The dotted line shows the $\lambda$ -transitions.	271
Figure 63. Computing the combination of labels that can be applied on an elementary 3-simplex. A total of 35 ways are possible.	272
Figure 64. The number sets generated by the java program based on the trial and error approach.	280
Figure 65. The relationship of $B(n)$ vs. $n$ . An exponential curve with the values shown in the figure gives the best fit.	284
Figure 66. A “trackback” that shows the dependency of the various problems that need to be solved before an algorithm can be devised to do an exhaustive search of a triangulated $n$ -simplex.	285

# Chapter 1

## Introduction

### 1.1. Agents and Multiagent systems

Computers pervade every aspect of human life today. In just a few decades, computers have evolved from a programmable calculator to being able to fly airplanes better than humans can. As computer hardware becomes more ubiquitous and cheap, the focus has shifted to developing better software. Consumers have become more demanding of what software is expected to do, as well as less tolerant of software failures. As we become more reliant on computers to take charge of the routine and dangerous tasks that need to be performed, we have become more susceptible to software failures. Software has become exponentially more complex to design, develop, and test. Software designers and developers are slowly coming to accept that the traditional model of developing a monolithic block of code is difficult to design and debug. The trend is now more towards developing distributed software architectures that allow for more robust and fault-tolerant behavior. Thus researchers are now focusing on designing software systems that are composed of mostly independent software entities that interact together to produce the desired behavior. These independent software entities are termed “software agents,” An agent is a computer system that is capable of independent (autonomous) action on behalf

of its user or owner [1]. The term “agent” covers a wide range of behavior and functionality. The more complex the considerations that the program takes into account, the more justified we are in considering it an “agent.” But we can still specify some general characteristics an agent would embody [2]:

1. An agent is persistent. Unlike function calls in procedural languages, an agent thread does not terminate once a call to it returns.
2. Agents are proactive. Normal software is reactive and has to be explicitly invoked. Agents, however can perceive, reason about, and initiate activities in their environment.
3. Agents are social. Typically, software is developed as a monolithic and isolated piece of code and restricts information transfer through function calls and messaging. Agents are capable of communicating with humans as well as other agents using higher-level distributed protocols like FIPA [3].

Agents are therefore a clear departure from the traditional paradigm of object-oriented design. They are being developed as the next generation of software architecture. There exists considerable literature on how individual agents can be designed using various architectures, such as ones based on belief-desire-intention (BDI) [4] etc. But beyond individual agents, the focus of research is now on understanding the group behavior of agents. This area of research is called “multiagent systems.”

While individual agents enhance the user interaction with computers and improve productivity, the real power of agent technology is utilized when agents are set up in a

social environment where they can interact with other agents. Multiagent systems have a number of useful attributes, such as decentralization of knowledge and computing power. Due to the high complexity of present day software, a multiagent approach makes the design and development of such software more tractable and easier to understand. Multiagent systems also tend to be redundant which makes software more fault-tolerant and robust. Sometimes centralized approaches may be impossible because the business logic and data may be the property of individual corporations, who may want to keep them private for competitive reasons. Agents can thus function as intelligent application programs that control the flow of information to the public web, while producing the desired behavior. To successfully interact, agents require the ability to cooperate, coordinate and negotiate with each other. Multiagent systems design can be decomposed into

- Individual agent design. This is the micro level.
- System design. This is the macro level

Generally, there is no central controller in open multiagent systems and since agents may represent their owner's interests, they should be assumed to be selfish, competitive, and capable of lying. These agent behaviors come into focus as electronic marketplaces get increasingly dominated by software rather than humans. Therefore there is a clear need on the micro-level that agents be designed to be clever enough to survive competitive effects. On the macro level, system designers should ensure that protocols are appropriately designed that induce an agent to be truthful, while keeping its data private. There has been intense focus therefore on the mechanism design for multiagent systems.

This is crucial because agents are being deployed in various forms over the internet. For example, shopbots [5] will frequently poll various websites to find out the lowest price available for a good. It is possible that shopbots can collaborate on the web to exchange price information, so that redundant searches are avoided. But for this, fair protocols must be put into place, so that a particular agent is not exploited.

Another area that will benefit from the agent oriented paradigm is Web services [6]. While Web service architecture has attractive features, such as decentralized control, and data privacy, they have not been designed to be proactive. Agents can be used to add more intelligence to Web services in the form of composing two or more individual web services and presenting it as a new Web service to the user. With agents working to coordinate Web services in the background, these compositions may be more robust to individual Web service shutdowns or failures. Agents can also improve the ease of use of Web service and may proactively go out to seek new consumers for its services. They can inform the consumers of updates and changes. Agents can use ontologies to compose web services on the fly and they may be able to negotiate better rates using high-level communication protocols like FIPA.

As can be seen from the above scenarios, conflicts will arise whenever there is demand for limited resources like computing power, storage space or Web services. To resolve these conflicts a suitable mechanism has to be set up. Negotiation is a powerful, general mechanism for resolving conflicts among autonomous agents. Multiagent systems consist of computer systems made up of machines that have been programmed by different

entities to pursue differing goals. This is distinct from the approach of distributed computer systems which have been centrally designed to pursue a single global goal. Rather the problem domain consists of looking at situations where computers belonging to distinct and (maybe) competitive agencies interact to control and share common resources.

I discuss the criteria that a good negotiation protocol should have in the next section.

## **1.2. Multiagent Negotiation**

Why do agents need to negotiate? Let us begin by setting up the context. Autonomous agents are currently being given responsibilities in numerous applications from the economic, industrial, commercial, social, and entertainment sectors of the world. Autonomy means that the agents have a high degree of freedom and choice in initiating actions on their own, planning goals for themselves, and taking actions to achieve them. In order to achieve their goals agents must have sufficient resources and capabilities. But what if the resources are insufficient for all agents to achieve their goals? This typically is the case where an agent lives in a multiagent system and has to share resources with other agents, coordinate its own tasks to resolve conflicts, and sometimes persuade other agents to do things for it. All this could be done centrally by one designer, who could chalk out every detail of the entire system. The central agency would then be able to resolve conflicts, divide resources, plan schedules, etc. But this is not applicable in open systems where individual agents are designed by different designers and there is no concept of a



central agency. Multiagent systems are inherently distributed, heterogeneous, and open. So what could be done in situations like this? In the real world, people from different backgrounds come together and resolve conflicts or form alliances for mutual benefits. This sort of interaction between people is based on negotiation. Negotiation [7] is “when two parties strike a deal through argumentation or arbitration for the mutual good of both.” Negotiation as per Davis and Smith [8] is “A discussion in which interested parties exchange information and come to an agreement.” This work revolves around a resource allocation problem and the main concern for agents participating in the negotiation scheme described is to ensure a fair and unbiased allocation of a resource. In open multiagent systems there is generally no global control, no globally consistent knowledge, and no globally shared goals or success criteria. So there exists a real competition among agents, which act to maximize their own utilities. I assume all the utility functions are private to the agents. The negotiation protocol should be immune to information hiding and lying by agents. This has to be ensured as there is no control on the design of agents that interact in open environments and the only check that could be made is by cleverly designing their interaction mechanism.

The discussion of protocol design here is on higher level than the typical protocols used for purposes like file transfer (FTP) or accessing HTML pages (HTTP). Those protocols deal more with the mechanics of information transfer and operate only when humans invoke it. I also do not deal with the higher-level issue of how agents will communicate [9] with each other. Agents are assumed to be able to communicate and share a semantic

understanding of the domain they operate in. It is possible that the agents refer to a common ontology to clarify what the terms in the negotiation mean [10].

Broadly the major features of a negotiation process are:

- The language used by the agents participating in the negotiation
- The negotiation protocol that the agents need to adhere to.
- The strategies adopted by each agent in order make the deal.

The issues can be looked at from the agent environment perspective. This basically falls in the sphere of mechanism design. Here I deal with the issue of designing appropriate negotiation protocols such that agents, irrespective of their origin, capabilities or intentions will interact productively and fairly. At the basic level the negotiation mechanism will be evaluated by the criteria of:

- Efficiency
- Stability
- Simplicity
- Distribution
- Symmetry

These are desirable features in any negotiation protocol, but real world protocols generally are able to satisfy only a small subset of them. The above list is not exhaustive. Additional criteria are analyzed in detail later in the text.

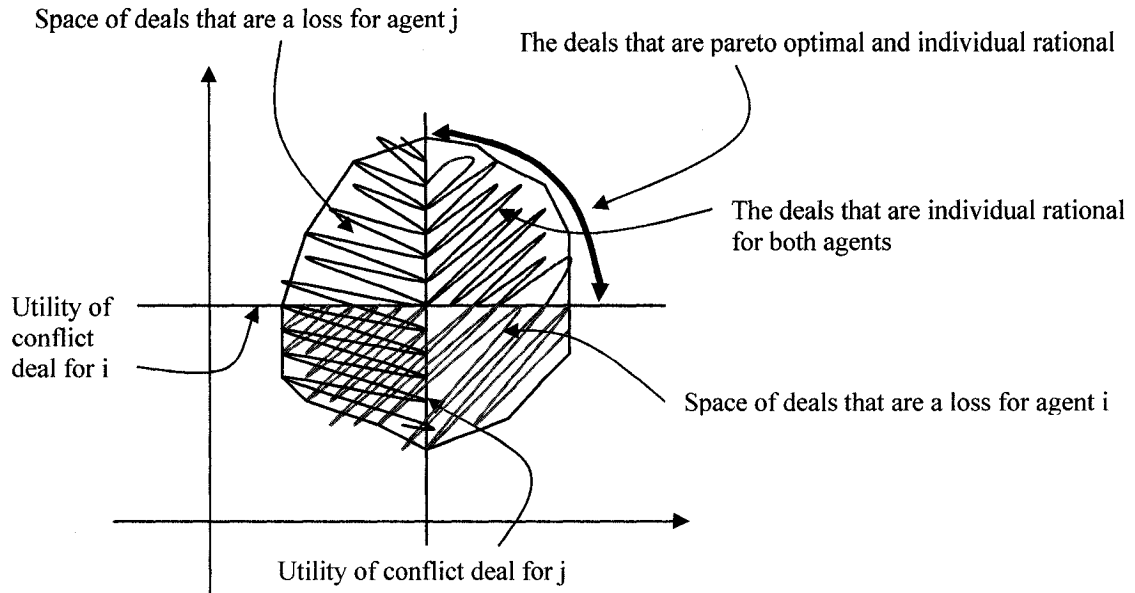
The protocol design can also be viewed from the individual agent's perspective, i.e., given an environment in which the agent operates, what is the best strategy for it to

follow? This falls in the sphere of strategy design. Negotiation strategies have been developed from this perspective, but they tend to be applicable only to specific problems and have been difficult to generalize to broader problem domains.

Negotiation language is an aspect that needs to be considered in mechanism design as well. For example, the various features of a negotiation contract, like the cost, quality of service, deadlines etc. can be specified in a standard XML protocol, whose semantics are universally understood by all agents. Thus the agent utility function should not only figure in the cost of the good being bought or sold, but also its quality, speed of delivery, penalties for breaking the contract, reliability of the opposing agent among other factors.

Any negotiation setting will have four components [1] :

- A negotiation set: This is the set of possible proposals that agents can make. The negotiation set is the set of deals that are individually rational and Pareto optimal.
- A protocol. This is the set of possible actions an agent can take during negotiation.
- Strategies. This is the plan of actions that each agent sets up for itself to get the best deal. This is private to each agent.
- A rule that determines when a deal has been struck and what the agreement deal is.



**Figure 1.** Space of possible deals for agents  $i$  and  $j$ . This is an elaboration of the figure drawn by Woolridge [1]. The polygon is used to represent any closed convex shaped region.

Game theory has been extensively applied to discuss negotiation protocol design [7]. It cannot be assumed *a priori* that agents will be cooperative, share information or sacrifice for greater good. By appropriately designing the rules of public behavior (rules of the game) by which agents interact, one can influence the private strategies that will be designed into the agents. Thus it is possible to create protocols that encourage efficient and stable behavior by the agents. Game theory provides a good toolset for interaction mechanism design because agent societies fulfill the assumptions of predictability, consistency, and narrowness of purpose. They can particularly fit the rationality assumption of game theory better than humans. A nice example of good mechanism design that induces truth telling and stability is the Vickrey auction [11]. Here agents are motivated to submit bids that reflect their true utility. The winner is the agent that submits

the lowest bid, but gets paid the price of the second lowest bid. Like any mechanism however there are always drawbacks. Agents may be discouraged to bid because they have to part with private information (i.e., their true valuation of the bid) and the auctioneer may not always be willing to pay out a higher price than is being asked for in the lowest bid, especially if the difference is substantial.

Three types of environments have been identified [7]. I describe them below.

### **1. Task Oriented Domain.**

A task-oriented domain is one where agents have a set of tasks to achieve, all resources needed to achieve the task are available, and the agents can achieve the tasks without help from each other. Each agent has a cost function for the list of tasks it can perform. It is assumed tasks do not interact with each other. Agents come together to see if they redistribute tasks in a way that is beneficial to both of them. The monotonic concession protocol has been proposed for negotiating in this domain. This consists of agents successively making concessions (in the form of deals) to each other until one of the following occurs:

- The other agent thinks the utility it gets from this deal is better than the one it got from its own offer and the negotiation terminates in an agreement.
- Neither agent concedes and the negotiation terminates in conflict.

It has been shown that if one agent follows the zeuthen strategy – whereby at every step it gives just enough concession so that the balance of risk is changed – then the best that the

other agent can do is to follow the same strategy [7]. The zeuthen strategy is however unstable at the last step and by adding in a mixed strategy to this, stability can be regained. This is called the “Extended zeuthen strategy.” The drawbacks of this strategy are:

- It requires agents to know each other utility functions, which is generally difficult in a non-cooperative environment.
- The players should know the entire negotiation set which is a computationally expensive  $O(2^n)$  task.
- The negotiation process may involve transferring messages over many steps that adds to the communication burden.

## **2. State Oriented Domain.**

State oriented domains (SOD) encompass a larger class of scenarios for multiagent encounters than Task oriented domains. Task oriented domains (TOD) are in fact proper subset of state oriented domains. Unlike TODs , in SODs it is possible for tasks to interact. The interaction can be positive or negative. This may happen because resources that agents need to perform tasks may not be unlimited like in TODs. Consider the mailman delivery problem; rather than assuming that agents have unlimited delivery vans to deliver mail, there may only be a single van available for both agents. If one agent is currently using the van, the other agent’s tasks are negatively affected because of non-availability of the van. In SODs, agents have to deal with global conflict and interference, as well as the possibility of unintended cooperation. Each agent strives to change the state environment so that it is in one of the goal states. One need not enumerate the goal states

explicitly; any state that adheres to a certain rule (all mails have been delivered) can fulfill the criteria of a goal state. Goals cannot be partially achieved, the condition of being in the goal state is atomic i.e the agent is either in the goal state or it is not. Since it is possible that an agent may perform any role in the joint plan, it becomes essential that the cost function is not predicated on the agent. This, however, can be an unrealistic assumption. The domain also does not cover possibilities of meta-goals (goals about goals). Another aspect that needs to be considered is reachability i.e., will the agents be able to reach their respective goal states while working cooperatively?

There are four possible interactions, from the point of view of the individual agent:

- A symmetric cooperative situation is one in which there exists a deal in the negotiation set that is preferred by both agents over achieving their goals alone.
- A symmetric compromise situation is one where there are individual rational deals for both agents that they prefer over goal states that can be achieved jointly, but the agents prefer the joint states rather than leaving the world in its initial state in case a conflict deal is reached.
- A non-symmetric cooperative/compromise situation is a deal that is seen as cooperative by one agent and as a compromise by the other.
- A conflict situation where no individual rational deals exist.

Rosenschein proposes the Unified Negotiation Protocol framework that uses product maximizing mechanisms to deal with conflict as well as cooperative situations.

### 3. Worth Oriented Domain.

In worth oriented domains the assumption that the state is either a goal state or not, is relaxed. Rather a worth function is defined over all possible final states. States that have the highest value of worth can be thought of as fulfilling the full goal, while those with lower worth values, only partially satisfy the goal. The worth function stands in place of and encapsulates the classical artificial intelligence notion of a goal as a set of acceptable final states. State oriented domains are a proper subset of worth oriented domains. Agents can thus compromise their goals to the extent needed to reach a deal while still getting positive utility. One can then define a measure of utility derived by an agent in terms of the “distance” of the current state from the goal state. The “distance” metric can be specific to the domain being looked at. For example, the value of a seat in a theatre is directly proportional to how close it is to the stage. There is also a notion of probabilistic distance when the relationship between the intermediate state and the final state is indeterministic and can be used as a possible worth function.

The protocols proposed for these domains can be classified as utility function based protocols. They require the agents to make decisions based on a utility function they follow. It is assumed that agent designers will want to maximize their expected utilities. This is not something that designers will necessarily adhere to. One also has to be careful that situations like the St. Petersburg paradox [12] do not arise where expected utility values can become infinite. Another drawback of this approach is that designers are expected to transform other agents’ utilities into common utility units. This is problematic for various reasons:



- Interpersonal comparisons of utility are suspect because there is no good way to interpret how different people value goods. Utility [13] is a measure of an individual's happiness gained by the consumption of goods and services. Utility is not directly correlated with money and hence will be difficult to measure in common units
- To know about other agents' utility function, either the designer must have good *a priori* knowledge of the other agent's design or all agent utility functions must be made public, which is a discourages adoption of the protocol.
- There is no guarantee that the monotonic concession protocol will terminate in a finite period of time. Both agents may make infinitesimally small concessions to each other keeping the process running forever. Rosenshein's protocols also do not apply when more than two agents are involved. For example would the monotonic concession protocol work for a general case of  $n$  agents? Can we still apply the Zeuthen strategy in that case?

### 1.3. Fair Division Schemes

Fair division schemes can be thought of as a sub-topic in multiagent resource allocation. They however pre-date multiagent theory by almost three millennia. This is because the essential issue of dispute resolution that is being solved for multiagent situations applies very analogously to person-to-person and even country-to-country disputes. There exists a rich source of literature on the subject of fair division, or the cake-cutting problem as it is more informally called. The subject touches many fields of study and hence has

benefited from contributions from mathematicians, economists, sociologists, political scientists, and, recently, computer scientists.

Why is the subject of fair division so appealing? One important reason for this is the possibility of self-mediation. While there are other means of settling disputes for resources, like auctions, legal settlements etc, these invariably find the need for a central authority to apportion resources. There are a number of problems associated with the approach. One problem is that of mediator bias, i.e., the mediator can team up with one of the disputants for a percentage of the profits to manipulate the results in a particular agent's favor. Since the approach involves the subjective evaluation of facts to decide the allocation, it can never be proven one way or another of the mediator's impartial intentions.

In the multiagent context, another problem is of centralization of command. This goes against the philosophy of robust distributed systems that are immune to specific system failure. For example, if in the middle of a bidding procedure for an internet auction, if the central server goes down, the entire multiagent system goes down. One can therefore appreciate the need for self-mediating protocols.

The cake-cutting problem can be most easily visualized by the following problem: A cake needs to be divided among two agents. What is methodology that can be followed so that the cake can be divided by the agents among themselves; such that each agent is satisfied it has received a fair share of the cake? An simple and elegant way of doing this

is by allowing one agent(divider) to divide the cake and letting the other agent(chooser) choose a piece for itself. The chooser has to be satisfied since it can evaluate and pick what it thinks is the larger among the two. The divider can't complain, since it cut the cake and both pieces should have exactly the same value in its perception. Note that no mediator or central agency is involved in apportioning the cake and hence the attendant problems do not exist. Unfortunately however such an elegant solution has proved difficult to scale up to more agents. Trying to scale up to scenarios where there are  $n$  agents in general is the focus of this dissertation. Briefly, it can be said that researchers have attempted to tackle specific aspects of the problem. Mathematicians have given measure-theoretic or combinatorial solutions but are limited in the useful criteria they can fulfill. Economists have proved existence results for  $n$  agents, but not many constructive solutions exist. Generally the constructive solutions come with caveats. Brams and Taylor [14] mention four criteria which have been used to evaluate the various algorithmic procedures proposed for cake cutting. They are:

- Proportionality: A procedure is proportional if each agent believes the value of the portion it has received is at least  $1/n$  of the total resource allocated.
- Envy-freeness: A procedure is envy-free if each agent believes the value of the portion it has received is at least as large as any other agents' portions.
- Equitability: A procedure is equitable if the goods are divided among two agents, such that both agents not only think they got a larger share, they also think their share is larger by the same amount.

- Efficiency: A procedure is efficient if it allocates the resource so that there exists no other allocation that makes every agent at least as well off while making at least one agent better off.

A deeper discussion of these properties is reserved for a later chapter. Brams and Taylor remark however that, in general, it is difficult for any algorithmic procedure that apportions resources for more than two agents to fulfill more than two of the above given properties at any time. Thus the simple “divide and choose” procedure for two agents is riddled with problems when one attempts to scale it up. It can be mentioned here that envy-freeness implies proportionality while the vice-versa does not necessarily hold. Thus envy-freeness is a stronger constraint than proportionality and consequently fewer algorithms manage to achieve it. Since scaling up to  $n$  agents is a difficult problem in itself, most procedures commonly manage to achieve proportionality and (less commonly) one of equitability and efficiency. The moving knife procedure is an elegant solution for  $n$  agents to divide a cake among themselves. But since the knife is held by a disinterested outsider, the requirement for a mediator creeps in. Thus it is difficult to classify the moving-knife as a purely self mediating solution. This shows that although the proposed procedures attempt to be self-mediating, mediation may be involved in subtler levels of implementation. It may well be that completely doing away with the mediator role might be impossible and such issues must be resolved at the time of procedure implementation.

Various aspects of negotiation and procedures for resource allocation are discussed in the next chapter. Chapter 3 discusses the various criteria that can be used to evaluate negotiation protocols and procedures. Chapter 4 then describes a novel procedure being proposed in this dissertation that attempts to solve the one-dimensional resource allocation problem for the  $n$  agent case. Chapter 5 tries to extend the essential principles of the solution to higher dimension resources. In Chapter 6, I explore the possibilities of a solution for a different topological surface than the flat two-dimensional Euclidean plane. I also discuss difficulties faced in solving this problem. Chapter 7 looks at the possibility of using the Sperner's Lemma property for solving fair division problems. Finally I end with a discussion on various issues involved in using topological approaches to solving multiagent resource allocation problems.

#### **1.4. Motivations**

Fair division (more informally cake cutting) has provoked an interest since ancient times. It was used typically in situation where a judgment was required to settle disputes. The oldest known example of the application of fair division is mentioned in (1 Kings 3:28) which tells the tale of two women coming to the court of King Solomon claiming motherhood over the same baby. The king's proposed solution of dividing the baby into two revealed the true utilities of the women, whereby one woman ceded the whole baby whereas the other woman was willing to accept the judgment. The truth of the baby's maternity thus became apparent. The history of cake cutting is filled with such interesting stories. The strategy, whereby one cuts the cake and the other chooses is known as the

“divide and choose” procedure. Variations of this theme have been applied to solve various historical problems.

Political theorist James Harrington [15] put forth a solution to divide power evenly among the two houses of legislature. He proposed that the Senate be given the necessary function of debating or dividing, since they consisted of the wisest and most virtuous members of society. The House on the other hand – being made up of representatives of the people and not expected to be particularly intelligent – would have the power to vote for or against the bill. Thereby Harrington argued that the vested interests of neither chamber would become dominant. This is generally how the mechanism of checks and balances works. Brams and Taylor call it the “Filter and Choose” procedure, whereby one agent has the ability to propose bills and the other the power to vote it into legislation. There is however nothing to be shared in this case. Both the chambers can lay claim to the same piece of cake i.e., the bill that was passed. The U.S. legislature is based on a similar bicameral format, but there are significant departures from what Harrington had suggested. In the US Congress there is no distinction between divider and chooser as both bodies can debate and pass their own version of the same bill. If it so happens, then a compromise bill is created by a compromise committee comprising members of both houses. Once the compromise bill is created then both bodies have the option of voting it either up or down and it cannot be modified any further. In this case the conference committee becomes the divider and the houses are choosers.

The Law of the Sea Treaty is a culmination of several United Nations conventions and provides various controls for the management of marine natural resources and pollution control. Based on this treaty the International Seabed Authority (ISA) was created to administer and control mineral resources in ocean floor areas beyond the limits of national jurisdiction. “The convention of the Law of the Sea, which went into effect in 1994, incorporates such a scheme to protect the interests of developing countries when a highly industrialized nation wants to mine a portion of the seabed underlying international waters. The country seeking to mine would divide that area into two portions. An independent agency representing the developing countries would then choose one of the two tracts, reserving it for future use.” [16]

Divorce proceedings are a good example of how cake cutting can be used to resolve disputes. Divorce tends to be a bitter and acrimonious affair between two parties even as expensive litigation costs start to accumulate. Another complication is that the assets to be divided generally cannot be modeled as a single continuous resource (like a cake). The properties to be distributed may include indivisible objects like a house, car, jewelry and furniture. A modified version of the divide and choose is proposed by Brams and Taylor in this case. They call it the Adjusted Winner process using which each party allocates points to each object of dispute with the condition that the points should total to 100. Then the procedure goes ahead to allocate objects based on the importance (as reflected in the assignment of points) of the objects in the eyes of the parties involved. This modified cake cutting has many benefits like fairness, envy-freeness and equitability.

Thus cake cutting approaches can be used to get very satisfactory results from difficult disputes in the real world.

Denoon and Brams [17] apply the Adjusted Winner (AW) process to solve the dispute of the Spratly Islands between China, Taiwan and four members of ASEAN (Association of Southeast Asian Nations). The major issues in the dispute are: sovereignty, economic development, freedom of passage and regional security. Although AW allocations are efficient, equitable and envy-free they also have some major drawbacks, viz. interpersonal utility comparisons, manipulable to lying and the requirement that resources be divisible.

In 1944, at the end of World War II, the allies decided to divide Germany into zones. At first, they were not able to come to an agreement about what would be the status of Berlin. Subsequently, it was decided to partition Berlin itself into zones even though the city fell 110 miles inside the Soviet zone. In cake cutting terminology although it ended up as a “trimming” in the division of Germany, it was far too valuable for the western allies to cede to the Soviets. The division of Germany and later Berlin itself was probably done by some form of cake cutting, though in an informal sense. This example also brings to the fore the general problem of land division. While land division is similar to cake cutting in the sense of being infinitely divisible, there are still differences because it may not be recombinable. The constituents to whom the land is being apportioned may strongly prefer contiguous pieces of land rather their getting portions scattered into many separate “islands”. In the case of a region of dispute being bordered by many countries



(each of whom want part of it), another strong preference may be that the resource so allocated be contiguous with the border of that country. Thus no country will prefer to get a portion (even if contiguous) such that a portion of the opponent's territory has to be traversed to get there. Berlin was an exception in this case, where the Western allies preferred to get a portion of Berlin even if it was non-contiguous with the rest of their allotments.

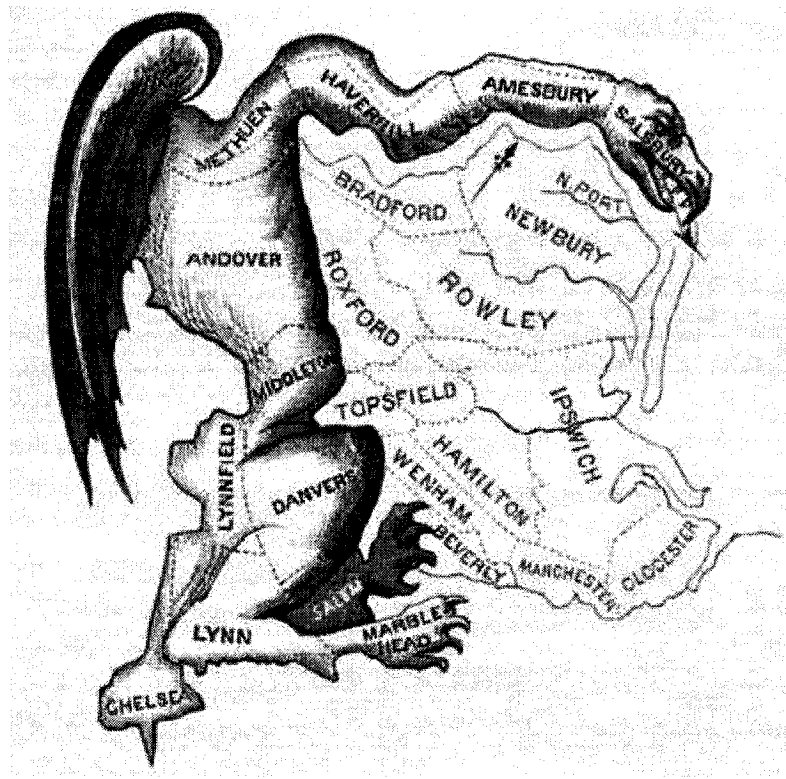
As computer storage grows leaps and bounds, it is getting cheaper to offer large storage space on the Internet [18]. Another trend is for putting autonomous agents on the Internet to do the owner's bidding. These two trends together create new scenarios where conflicts for resources may arise. Consider the case where a company offers its employees 10 terabytes of storage without setting strict limitations per user. This may be because different employees may have different storage patterns and hence it is expected that the employees will negotiate among themselves to reach a satisfactory solution. A moving knife protocol which allows employees' agents to bid for space seems to offer a good solution to this problem. A similar issue arises when a single high bandwidth connection is shared by many users (or their agents) in the same house. Based on their usage patterns a moving knife protocol or a multidimensional resource allocation protocol from cake cutting domains can offer a good solution to settle disputes amicably.

Cake cutting approaches are also useful in dividing undesirables among people or agents analogous to how desirables are divided. This is a dual problem to the problem of cake cutting. Martin Gardner [19] is apparently the first person to interpret fair division in

terms of setting up chores like mowing the lawn or clearing the garbage. Here each player would like to receive the smallest piece of the set of chores. While this problem is probably as old as cake cutting itself, it has received far less attention from researchers than cake cutting. One reason is that since the problem appears the inverse of cake cutting in many ways, its solution was always seen as an extension of cake cutting solutions. Another example of this dual problem is partitioning of rents in a household. Consider a group of housemates moving to a house with rooms of various sizes and features. Each housemate values every room differently and is willing to contribute to the rent according to that valuation. The question is: Is there a way to partition the rent so that each person will prefer a different room? Su [20] has proposed an elegant approximately fair solution that uses a fundamental lemma of combinatorial topology called Sperner's lemma. This solution has takes it's inspiration from cake cutting solutions. The traditional divide and choose procedure also works very well if there are just 2 agents involved. Other solutions include using the moving knife, but this time agents are awarded pieces to the right of the cake (rather than the left). This resembles the Dutch auction where the agents willing to bear the largest amount of chores bids first. It is rational for the agent to call when it feels that the (reducing) task is not more than  $1/n$  of the whole set. By bidding earlier, it may have to do more than its fair share. However if the agent waits till later (in order to get less work), another agent may step in to take this portion and the first agent might end up with a portion later in the allocation, that is bigger than the current portion. Hence it is to the benefit of the agent that is calls out when it gets exactly  $1/n^{\text{th}}$  portion of the whole.

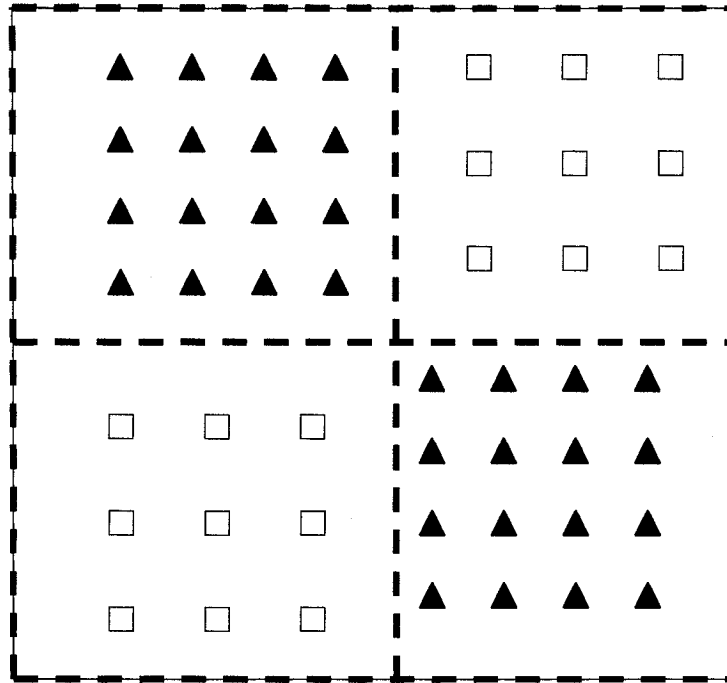
Inheritance disputes are similar in nature to divorce procedures. The problem is to apportion common inheritance among all the heirs such that everyone feels satisfied that the division was fair in their eyes. Unlike divorce procedures however, there may be more than two parties involved in the dispute. Also unlike traditional cakes, the objects to be divided may be indivisible. This problem is therefore more complicated than divorce proceedings. A point allocation procedure similar to Adjusted Winner exists. It is possible for each heir to get an envy-free allocation based on such procedures, but some compromises may need to be made. For example, if there is an indivisible object like a house of great value to all the agents, then it will have to be made “trimmable” by selling the house and using the money as proxy instead. Also each heir will have allot points to object representing its value in their eyes. This information can be used by competing heirs to manipulate their point allocations in such a manner that can give it more than the fair share. Nevertheless it is a good starting point to find a better solution that is less vulnerable to manipulation and offers a fairer deal to all heirs.

Gerrymandering is form of redistricting in which the electoral district or constituency boundaries are redrawn to advantage one political party over another [21]. The term originated from back in 1812, when Elbridge Gerry, as governor of Massachusetts, presided over the creation of a tortuous electoral district that resembled a salamander in shape. Gerrymander is thus the fusion of two words used to reference the dubious process of redistricting for electoral gain.



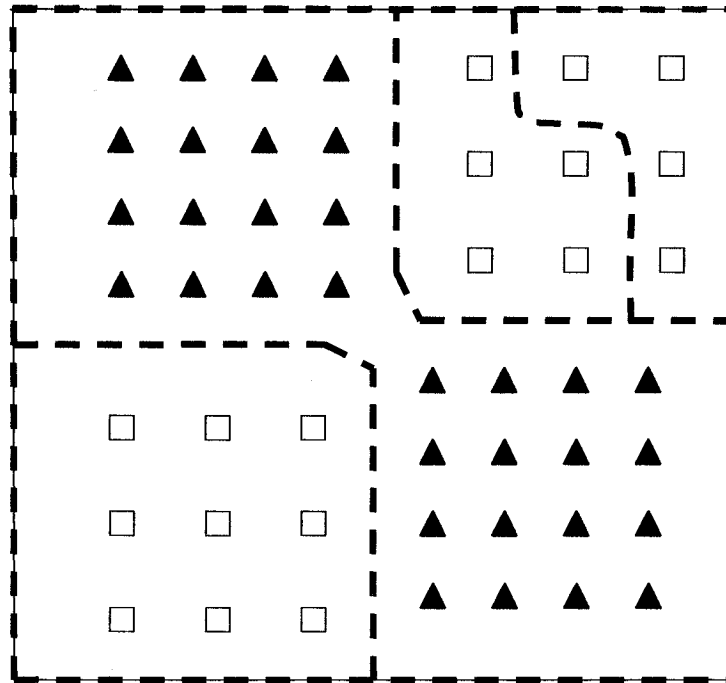
**Figure 2.** A political cartoon illustrating the electoral districts drawn by the Massachusetts legislature to favour the incumbent Democratic-Republican party candidates. The image is in public domain.

All electoral systems that use district level elections to determine representations are vulnerable to gerrymandering. Gerrymandering is most effective when the system of voting is nonproportional i.e., the number of people voting for a particular party does not translate into a proportional number of seats in the House. This typically happens when “first past the post,” is used as a criterion for selecting the winner among competing candidates. By this criteria, the candidate with the maximum number of votes wins.



**Figure 3.** A standard drawing of districts ( dashed lines). The squares and the triangles represent voters with different party affiliations. A total of four districts need to be drawn. Thus two districts will be won by the triangle party and two by the square party.

In such scenarios, gerrymandering can be used to get the wasted vote effect, i.e., if a candidate is guaranteed to win in a particular district, then opponents can redraw the district in such a manner so as to stuff as many supporters of the candidate as possible into that district. While this does not improve that candidates chances of getting elected (since he would have won anyway), it improves the chances of candidates standing for the rival party in adjoining districts to get elected. This is because they have effected a shift in the political demographic of the population, thus having a better chance of winning. This is known as “packing.” (Refer to Figure 4).

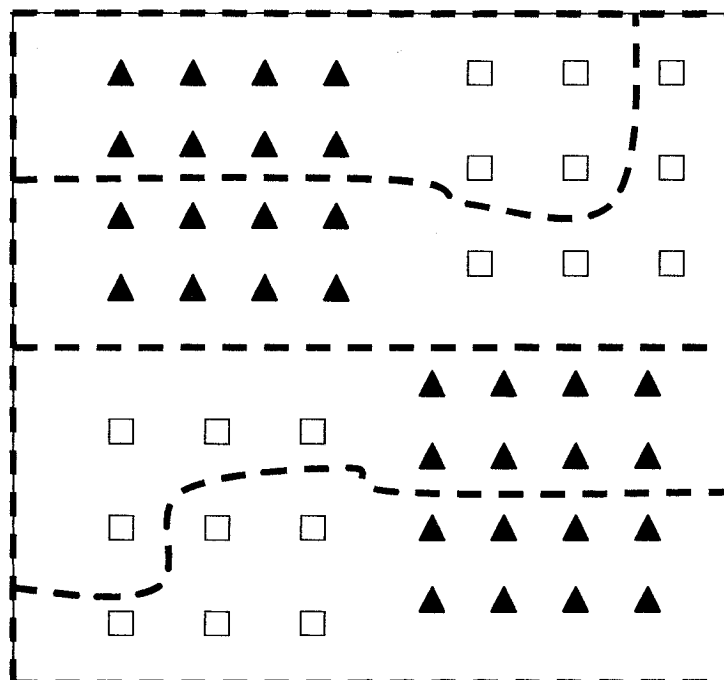


**Figure 4.** A representation of how “packing” is done. Since the voting mechanism is nonproportional, all constituents of the triangle party can be packed into one district, while the remaining districts go to the square party. In this case squares get an advantage by winning three seats out of four”

“Cracking” can be thought of as the inverse. Here if a particular district has a majority of voters casting their ballot for a particular party, the opponents can redistrict that block and scatter them among adjoining constituencies, thus diluting their voting powers. Refer Figure 5. The “sweetheart” gerrymander is a redistricting strategy where rival parties get together in order to further strengthen their own voting blocs and guarantee their election into power. While this plan benefits both parties, the electorate tends to lose out because of reduced competitiveness among candidates. Since candidates are assured of winning, they tend to lose interest in campaigning. Campaigning costs are cut down and candidates may be less motivated to produce results for their constituents (better roads, civic

amenities etc) as they know they will get elected anyway. This has been a historical source of Congressional acrimony every 10 years since 1790. There has been plenty of debate as to how the redistricting process can be made less controversial. It becomes quite clear that computers [22] can play an important role in this process. In order to write a good algorithm though, the criteria of what constitutes a good district needs to be fixed first. Commonly agreed upon criteria include:

- Contiguity: Each district should be a single contiguous territory.
- Compactness: The metric to be used to define compactness itself has been a tricky procedure. Many definitions of compactness while seeming to be intuitive can still yield geometric curiosities that can allow gerrymandering.



**Figure 5.** A representation of how “cracking” is done. Here the triangle party redraws the districts in such a way that it breaks up square majority areas. With a nonproportional system, a simple majority is sufficient for a candidate to win. Due to this all four districts are won by the triangle party.

- Homogeneity: Districts should represent communities of interest. Thus rural communities will have distinct preferences from urban ones and the redistricting plan should reflect these issues.
- Continuity: Constituents may be more willing to accept a redistricting if still bears some resemblance to the previous version. Any sudden or sharp changes may arouse people's suspicion and should not be allowed.

This is in no way an exhaustive list. But all these properties can be suitably set up in a computer in order to check the effectiveness of algorithms creating districts. Cake cutting has been suggested as an option for such a solution. For example, one can assume a knife moving from east to west (or any arbitrary direction) parallel to itself and cutting when about  $1/n^{\text{th}}$  of the population has been covered (if  $n$  districts are to be created in a state). This procedure fulfills all the previous properties stated above (except homogeneity). But it can create geographical curiosities by creating some long slender cuts of land that may not look very appealing. It however becomes clear that enabling computers to do the job will reduce the controversy and redistricting may actually start to help in improving competition and representing the aspirations of the electorate better. As it stands however computers are being extensively used by individual parties to gerrymander even more effectively by being able to zoom in to the level of an individual census block and sharpen the advantages to the incumbents to get re-elected.



# Chapter 2

## Literature Survey

### 2.1. Generalized Multiagent Negotiation

The literature on multiagent negotiations protocols is rich and varied. I briefly provide an overview of the various aspects of multiagent systems being focused on by researchers. Then I turn to the specific field of cake cutting domains, and analyse the contributions to that field. Negotiation, in order to enable multiagent collaboration for distributed problem solving has been discussed by Lesser and Durfee. [23]. These protocols are set up specifically for the domain of cooperative multiagent frameworks, where systems of distributed sensors collect and interpret data and finally reach a global solution. Durfee's recent work [24] includes trying to reduce the number of potential interactions between agents as the number of agents in the system increases. Potentially the number of interactions between agents increases exponentially with respect to the number of agents, so a solution proposed for this is to enable agent congregations where like minded agents get together in smaller groups that represent more specific interests. The number of interactions then scales better, as the number of interactions increase with the size of the congregation not the entire multiagent system itself.

Sycara has studied multi-attribute negotiation issues that consider case-based reasoning. Sycara has looked at how agents can negotiate in a manner where the process of negotiation can induce other agents to change their goals. Sycara [25] also focuses on bilateral negotiations between agents when dynamic and uncertain outside options are available for consideration. This situation is commonly encountered in commodity and service markets. Accepting one proposal means refusing all outside options. An example of such a scenario is a buyer concurrently negotiating with multiple sellers. Their model is composed of 3 modules, each of which encompasses increasingly sophisticated and complex scenarios than the former. The first module consists of single threaded negotiations that do not consider outside options. The second module consists of synchronized multithreaded negotiations which consider the impact of other concurrent negotiations taking place. The third and the most sophisticated model consists of dynamic multithreaded negotiations that take into account outside options coming in the future. The results of the experiments show that average utility obtained is higher as the sophistication of the models increases, i.e., if outside options are considered, the agents profit more from the negotiation. What is however not known is whether better results can still be obtained if other agents also consider outside options. The problem is that modeling the opponent who considers outside options is a non-trivial task.

Time can also be an important parameter in negotiation scenarios. This aspect has been explored by Kraus et al [26]. Negotiation time can be seen as a cost to agents and agents may discount the value of the deal reached based on the time taken to come to an agreement. Thus it may not be optimal for agents to haggle endlessly to get the best value

if it takes too much time. In fact, they show that the same agreements can be reached without delay and the resulting inefficiency is avoidable.

While the initial study of single agents was to model and automate some aspects of human behavior (searching for resources on the Web, scheduling appointments, etc.), multiagent systems have become a research area in their own right. Lin and Kraus [27] have refocused on the issues of automated agents for negotiation. They try to determine if automated agents can beat their human counterparts in getting better utility for the side they represent. These negotiations have multiple issues to be optimized within a given deadline. The humans are modeled as agents with bounded rationality and a qualitative decision making component is used. There is incomplete information about the reasoning used by the opponents. The qualitative decision making component can take into account the agent's utility function as well believed type of the opponent. The results were encouraging but mixed. While the agents played one side, they performed significantly better than the human opponents. But when they played the other side, their performance was not better than the humans. Thus this methodology of negotiation is useful. However it is not clear if the mixed results were because of side-specific issues (different starting resources, different time deadlines etc.) or if the probabilistic modeling needed further refining.

Fatima et al. [28] propose procedures for solving multi-issue negotiation under deadlines. The package deal procedure bundles and discusses all issues together whereas in the simultaneous procedure the issues are discussed simultaneously but independently of

each other. Lastly, the sequential procedure discusses issues one after the other. The authors rank the various procedures in terms of computational complexity, efficiency and run times.

Rosenschein and Zlotkin [7] present a rigorous study of multiagent negotiation protocol design and analysis. Their unique contribution is to show how analytical techniques and modeling methods from the world of game theory and decision analysis can be applied to the dynamic organization of autonomous intelligent systems. The book describes a theory of high-level protocol design that can be used to constrain manipulation and harness the potential of automated negotiation and coordination strategies to attain more effective interaction among machines that have been programmed by different entities to pursue different goals. It has already been mentioned in chapter 1 in greater detail of the various domains analyzed by the authors. In brief, they are the Task oriented domains, State oriented domains and the Worth oriented domains. A detailed analysis of how interaction mechanisms for each of these domains should be designed is described in the book. For Task oriented domains, the monotonic concession protocol was proposed and it was shown that the Zeuthen strategies adopted by agents will offer a deal that is pareto optimal. The extended zeuthen strategy was shown to bring equilibrium to the final step of the deal making and was in symmetric nash equilibrium. The authors take various simple domains like the postmen domain (where a task consists of delivering letters to mailboxes) to discuss the manipulability of the protocols. In order to obtain more utility the agents can lie by either hiding tasks, or stating to have more tasks (phantom tasks and decoy tasks) than they really have. Task oriented domains can be further broken down

into subadditive domains, concave domains and modular domains. The latter are a proper subset of the former. A rigorous analysis of the various combinations of lies, deals and domains throws up appropriate strategies for the agents to effectively thwart the opponent's design.

Although the analysis is thorough, there are still many aspects of multiagent systems which have not been covered. For example, the entire work assumes bilateral negotiation in a world inhabited by 2 agents. This may not be a realistic assumption. Generally any number of agents could be present in the world and there is a significant possibility of multilateral negotiation protocols being needed. Similarly the negotiation mechanism is only for one issue to be resolved. Again, in the real world, agents may need to satisfactorily negotiate multiple issues with each other. Time is also not considered as a criterion for negotiation protocols. For example the monotonic concession protocol may involve each agent making infinitesimally small concessions, thus taking forever to come to a conclusion. Nevertheless this book attempts to put negotiation analysis on a firm foundation and sets the stage for future research to build on.

Recently Rosenschein and Kraus [29] have concentrated on finding focal points among alternative solutions. Focal points refer to the prominent solutions of an interaction to which agents are drawn. They eschew the traditional negotiation models available to agents. Rather they assume that communication is expensive relative to computation. Thus in traditional scenarios where communication is used to coordinate agent actions, here coordination is attempted by making agents use contextual information to pick a

solution from multiple solutions in equilibrium. A simple game which embodies this principle is one where agents are told to divide 100 identical objects into two piles with the condition that they get positive payoff if they divide it just like the other agent would. Such games are called by game theoreticians as matching games, in which two players get a payoff if and only if both choose the same act; and the payoff is the same whatever the act may be. Thus this particular game has 101 possibilities where agents can come away with positive payoffs. It has been found that people traditionally choose an even division of 50 objects per pile. Agents trained in game theory and taught to behave rationally would generally choose any one of the 101 equilibrium strategies randomly and hence may coordinate poorly in communication-impooverished situations. The reason for choosing the 50-50 split among all equivalent solutions is that this particular solution has the unique property of being symmetrical and hence “stands out” with respect to the alternatives. Unfortunately, traditional formal representation techniques like those of the theory of rational choice where players choose their strategies on the basis of perceived difference in payoffs cannot capture why people are consistently drawn to a particular solution among many equivalent ones. Intuitively a few properties can be filtered out which may make particular solutions stand out. Some such properties suggested in the paper are: uniqueness, uniqueness complement, centrality, extremeness. This is not an exhaustive list. Identifying a focal point involves two parts: a structure that represents the players’ formalization of the game situation and a mechanism to derive a salient option from this structure. The paper attempts to provide general frameworks for building such discovery into an agent. Two approaches are suggested in order to locate focal points. The first is the decision theory approach. In the decision theoretic model, the agents are

assumed to be able assign utilities to various outcomes. An agent attempting to decide on an action using a decision theoretic framework constructs a decision tree, leading to different outcomes, with probabilities associated with each branch of the tree. The agent's expected payoff is the probability on the branch times the value at the leaf. The second approach is the logical focal point search. It consists of agent continually looking for candidates in the domain that have certain properties. If something in the domain has that property, it is a focal point at that particular point of time. As time goes on, new beliefs are derived and the domain over which the search is being conducted also expands. Both approaches have their advantages. Decision theory allows for a natural integration of payoffs into the decision-making process. The step logic approach is useful when agents act in dynamic worlds. In such situations the search for focal point has to be done where agents do not have all the information in advance. The technique is particularly well suited for modeling the time-dependent nature of focal point search.

There are number of issues regarding focal point search. This topic models itself after humans who are capable of sophisticated coordination with little explicit communication. What is inherently assumed is the cultural context that humans normally share. Humans generally go through a period of intense communication before communication gets cut off due to environmental reasons. Also if communication impoverishment is due to competitive reasons, an opponent may still get into an advantageous position by studying the cultural context of the agents and decipher focal points similarly. Although the paper tries to discuss focal points in an abstract way, it is quite clear that humans choose focal points based on the specific context in hand. Agents may thus need to be programmed

with the same cultural context as people. This is not only difficult but unnecessary. In a pure multiagent world, the focal points may be different from the one humans might look at. Nevertheless the paper provides an original counterpoint to the traditional negotiation based coordination.

Tohme [30] looks into the properties of negotiation that affect the quality of the resource allocation process. Most economic-theoretic approaches to the problem of resource allocation make little if any reference to the negotiation processes. Tohme looks at two models of economic exchange to glean the normative properties of negotiation. The Walrasian model assumes that agents do not directly interact with each other. Each agent starts in with an endowment and desires commodities belonging to other agents. A global price is set up for each commodity. The agents then pick commodities that maximize their utilities and offer their endowments in return. Communication is unnecessary since the agents only need to know the price which is globally available. Thus in game theory, the Walrasian model is set up as a non-cooperative game. The other model is the Edgeworthian approach in which agent interactions are allowed. A larger variety of feasible outcomes is possible in this model. The outcomes are determined by multilateral bargaining among agents. The edgeworthian approach can be seen as an instance of the general game theoretic problem of determining the outcome of a bargaining process. The author provides a detailed set theoretic explanation of both models in the paper and sets up the framework for analyzing negotiation processes. Tohme defines negotiation for resource allocation as “an iterative process of exchange of messages in such a way that agents change their characteristics and beliefs according to the messages they receive.” It



is assumed that procedural rationality applies, i.e., there is more emphasis on economic processes instead of economic outcomes. The paper sets up a comprehensive set theoretic framework for analyzing negotiation processes. It goes on to elucidate the normative properties of a negotiation protocol. For example, what should be the property of the set of messages (used for offer/counter-offer) in order to finally end in a stationary message? The author goes on to show that stationary messages signify the fixed point of the set and those points correspond to the status where global demand and supply of commodities are equal. This leads to the fact that the ensuing solution will satisfy pareto optimality. The agents are supposed to be capable of attaining a stable set of beliefs (after a finite number of iterations) and then based on those beliefs, emit a message requesting redistribution of endowments, giving everyone the best possible outcome according to a final stable preferential ordering.

## **2.2. Cake Cutting Protocols**

### **2.2.1. One-Dimensional Resource Sharing**

Krumke et al [31] focus on how a cake can be cut fairly making a minimal number of cuts. They can be classified into the scheme of approximate fair division algorithms. For this, they define a more flexible concept of fairness. A procedure is called  $\beta$ -fair, if every agent having a  $\beta$ -strategy can be guaranteed at least a fraction  $\beta$  of the cake according to its own measure, where  $\beta$  is a real valued number between 0 and 1. Since the number of cuts is considered an important parameter, the authors define what is called “frugal”

protocols. A protocol for  $n$  players is called *frugal* if it uses  $n-1$  cuts to complete the resource allocation. Dividing a cake fairly using the minimal number of cuts is impossible for  $n \geq 3$  players.

Even and Paz [32] have presented perfectly fair protocols for 4 players, making at most 4 cuts. Webb[33] discusses a perfectly fair protocol for  $n=5$  players with 6 cuts and shows that no perfectly fair protocol exists that uses only 5 cuts. Given that a frugal protocol is necessary, Even and Paz try to determine what is the largest value of  $\beta$ , for which there exists a  $\beta$ -fair frugal protocol for  $n \geq 2$  players. They prove that one cannot do better than  $\beta = 1/(2n-2)$ . While it is true that the number of cuts is an important parameter to be considered for calculating computational complexity, in general any protocol requiring  $O(n)$  cuts to divide a resource fairly should be quite satisfactory. The best that has been seen so far is  $n \log_2 n$ , but even that comes with caveats. Reducing the number of cuts by compromising the fairness criterion is not a good idea as this is the minimum necessary condition any rational agent will expect, to participate in a negotiation. The concept of getting a  $\beta$ -fair share of the resource might actually make the protocol look “unfair.”

Peterson and Su [34] focus on finding allocation procedures for chores. Chores are “bads” (rather than goods) that require allocation among agents. This problem is dual to the typical problem of cake cutting. However much less work has been done to develop algorithms for chore division than for cake cutting. The main reason for the gap is that chore division has been considered a straightforward extension of cake cutting methods. Typically agents will try to minimize the portions of tasks allocated to them rather than

maximize portions as in typical cake cutting. Thus an envy-free allocation of chores would require that each agent value its portion as smaller than anyone else's. For a 3-person case, the paper proposes a clever combination of Stromquist's [35] 3 person envy-free cake division procedure and Austin's [36] exact moving knife procedure for  $n$  persons to get a 3 person exact moving knife procedure. First using the Stromquist procedure, they model the chores as a cake, to create 3 portions of the cake such that every agent thinks a unique piece is the largest. Then the agent takes the respective largest piece and uses Austin's procedure to distribute the two halves to the other two agents. It similarly receives 2 pieces, one each from each of the other agents. The resulting allocation is envy-free.

Next, they propose a 4-person exact moving knife procedure which is inspired by the Brams-Taylor-Zwicker [37] 4-person envy free moving knife schemes for cakes. This requires 16 cuts in the worst case. Agents 1 and 2 use Austin's procedure to create four equal portions (in their respective valuations). Next, agents 3 and 4 suggest trimmings of the 3 largest pieces. Peterson and Su show that for various possibilities of the trimmings, every agent still ends up getting portions that it thinks is the smallest. Finally, they propose an  $n$ -person chore division scheme. It closely mirrors Brams and Taylor's  $n$ -person envy free cake cutting procedure. Rather than trimming to create ties for the largest portions (in cake cutting), they suggest "adding" to create ties for the smallest (in chore division). The addition requires reserves to draw on. These reserves are created by trimming before beginning division. The trimming and cutting procedures are however costly in terms of number of cuts ( $n^2+4n+2$ ). Also the  $n$  person algorithm may take

arbitrarily long depending on player preferences. There is no guarantee that this procedure will finish in a bounded number of steps.

Ismail, Busch et al [38] deal with open questions about cake cutting algorithms. They attempt to determine if the minimum number of comparisons required to divide the cake fairly can be fixed. They use an ingenious methodology whereby they reduce the sorting problem to cake cutting i.e., any fair cake cutting algorithm can be converted to an equivalent one that can sort an arbitrary sequence of distinct positive integers. This means that cake cutting is at least as hard as sorting. In order to do this, they propose the following procedure. The cake cutting algorithm is converted to solid piece algorithm and then to a labeled algorithm. Next they set up the utility functions of the agents such that their valuations reflect a positive distinct integer value. The set of utility functions is supplied to the cake cutting algorithm, which outputs a division of the cake into  $n$  portions, such that each agent gets a fair share. The ordering of the pieces from left to right is equivalent to the ordering of the numbers from right to left. Thus they can use (almost) any cake cutting algorithm to sort numbers. This enables them to show that (almost) any cake cutting algorithm will require  $\Omega(n \log n)$  comparisons in the worst case. This result is significant because it finds the complexity of the allocation for (almost) the entire class of cake cutting algorithms. Similarly they find that the lower bound for strong and super envy free divisions (if they exist) is  $\Omega(n^2)$  in the worst case. An important question is: Do the procedures for transforming the cake cutting algorithm to a sorting problem affect the complexity? The authors answer in the negative. They show that the extra cost of labeling is linear in the number of cuts. Essentially the

converted algorithm is as efficient as the original one. An additional requirement for the above result to hold is that the algorithms be comparison-bounded i.e., the computation that the mediator performs can only grow linearly with the number of cuts performed. While the authors verify that all known cake cutting algorithms are linearly labeled and comparison-bounded, they are unable to offer a formal proof that every cake cutting algorithm will fulfill these properties. Nevertheless, they provide the first concrete step towards proving the conjecture that no algorithm can do better than  $O(n \log n)$ . Note that the moving knife is a special case where, the algorithm is continuous i.e., it cannot be simulated by any sequence of discrete cuts. Thus the types of algorithms included in their framework are discrete or finite algorithms.

Sgall and Woeginger [39] determine the complexity of cake cutting using the number of cuts as a measure. This is in contrast to the work of Ismail, Busch et al [38] which treated the number of comparisons made as the measure of cake-cutting complexity. Sgall and Woeginger set up the following restrictions on the cake cutting model:

- Each player will receive a single subinterval of the cake
- The complexity is counted based on the number of evaluation queries as well as the number of cuts made.

Based on these restrictions they show that every fair cake cutting protocol for  $n$  agents will use  $\Omega(n \log n)$  cuts in the worst case. The results are extended in their later work [40] where they show a cake division scheme that guarantees at least  $(1-\epsilon)/n$  of the cake for each agent using  $O(n)$  cuts.

Edmonds and Pruhs [41] improve upon this work by providing an  $\Omega(n \log n)$  lower bound on the complexity of any deterministic protocol in the standard model of cake cutting. They relax the restriction on Sgall and Woeginger's cake cutting model that required each player to get a single continuous subinterval. The protocol is allowed to assign pieces to players that may be a union of disjoint intervals. The authors are able to extend the analysis of the  $\Omega(n \log n)$  lower bound to include randomized protocols.

### **2.2.2. *Hyperdimensional Resource Sharing***

The literature survey shows that the hyperdimensional resource allocation problem has not been discussed directly. Rather, it has been studied as a two-dimensional resource allocation problem in the form of land division. The two-dimensional resource allocation problem has been tackled by researchers from diverse fields who have encountered it in various forms. The literature survey describes researchers using the examples of dividing pizzas, cakes, or land to discuss the existence of fair solutions. Generally, however, the problems tend to have characteristics distinct from the one being solved here. One has not come across any constructive solutions so far to the generic land division problem. All the papers have offered existential solutions to qualified versions of the problem.

Hill [42] was one of the first to tackle fairness issues in land division. The problem domain was qualified because it attempted to allocate portions of land to countries that shared a border with it, such that each country received a portion connected to its borders. He extends a nonconstructive result proposed by Dubins and Spanier [43]. The problem

with the proposal of Dubins and Spanier was that although it could create fair shares for all agents, the pieces might not be in the neighborhood of the country to which the portion is allocated. Hill's solution is to create strips of land, thin enough, which connect the isolated portions, to the country it is allotted for. The strips created in this manner do not intersect because any pair of points in the set is assumed to be path-connected. There are some drawbacks to this approach. It can happen in reality that the strips that get created may be exceedingly thin so as to render the solution useless. For example, consider two countries that contest the land bordering them. It may be unacceptable for one country to have a single road connecting to its allocated portion surrounded by the enemy portions. Besides, this result does not explicitly provide a procedure for enabling such an allocation.

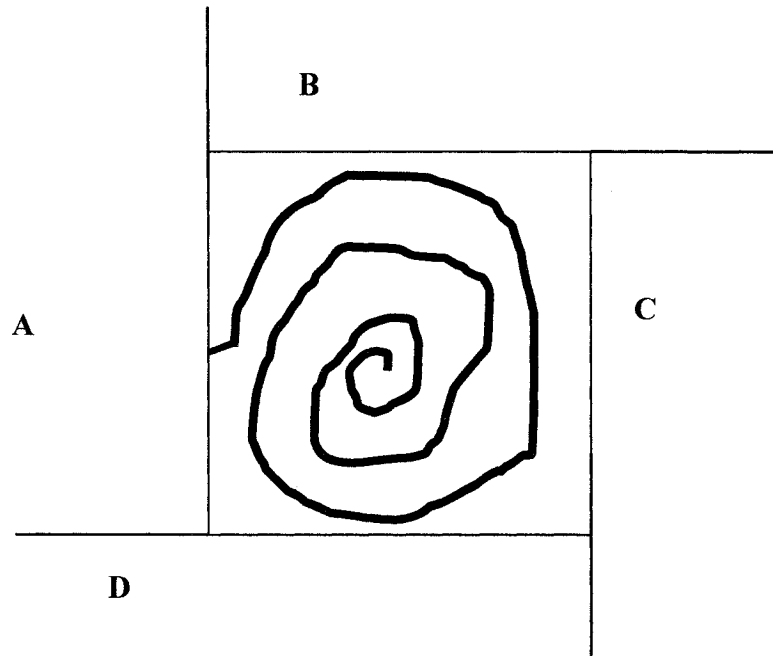
Beck [44] proposes a semi-constructive result that improves upon Hill's paper. In order to do this he constructs a unit disc  $|z| < 1$  that is a homeomorphic map of  $D$ , the disputed territory. It is also assumed that individual circles and radial lines in the disc have zero measure. Next, various agents place "bids" in an "auction" submitting the smallest radius that will enclose a disc, which is valued at  $1/n$  for that particular agent. The smallest such radius among the various bids is picked and the disc is awarded to that agent. In order that the portion may be connected to the agent's territory, a small wedge from the unit disc is also allotted. Then successive agents keep trimming the wedge so that the total piece allocated is less than  $1/n$  for each of the remaining agents. In order that allotted pieces do not end up having strips that break up other agents' allotments, a complicated procedure of secondary auctions involving rebidding of the same piece along with

guidelines for trimming and growing the pieces make it difficult to implement any such scheme. Besides, the solution is still hobbled by the issues affecting Hill's proposition. In fact, there may be some extra implementation issues, like getting the appropriate functions for the conformal maps, which make it just as impractical as Hill's procedure. These functions depend on the shape of the land and hence have to be tailor made for each problem individually. Beck only proves the existence of such functions, and does not specify how they can be found.

Webb [45] provides a combinatorial algorithm for the fair border problem based on Hill's existence results. The algorithm is recursive in nature and works as follows: A region  $R$  is bordered on all sides by  $n$  countries,  $C_1, C_2, \dots, C_n$ . Each country has its own evaluation of the piece of land and draws a region  $R_i$  adjacent to itself such it is valued at  $1/n$  by its own measure. Then each country in turn (that values  $R_i$  greater than  $1/n$ ) trims off a piece of  $R_i$  so as to disconnect it from  $C_i$ . The country that trims it last gets to keep the modified  $R_i$ . This region is attached to the allocated country by a strip of land small enough to be negligible to the others. The remaining land is then redistributed among the remaining agents similarly. The flexibility given by this algorithm in allowing agents to shape the region of interest as they like is also its drawback. If the shape of the land an agent gets is included in the fairness criterion then, earlier agents get a better deal than the later. This is because the later agents will have to "draw around" the regions allocated to earlier agents and the effective shape of the land they get might make it worthless for any use. In Figure 6, consider a square shaped region that is surrounded on 4 sides by 4 countries. Suppose agent A got the first chance to draw its region and it does so in the



manner shown in Figure 6. It can be easily seen how the other agents' allotments are placed at a disadvantage.



**Figure 6.** Agent A's demarcates of its share of the region. The thick line represents the land allocated to A

Hill's paper provides the basis for another result by Maccheroni and Marinacci [46] that proves fair border solutions exist even if the utilities are concave. Utilities tend to be concave rather than additive in the real world. They take into account the fact that the marginal utility of a good decreases as consume more of it is consumed, due to satiation. Classic cake cutting algorithms have always assumed that utilities were some sort of probability measure [47]. So utility functions had the following property:

$$v(A \cup B) = v(A) + v(B) \text{ for all disjoint } A, B \in \Sigma$$

On the other hand, a concave capacity has the property:

$$v(A \cup B) \leq v(A) + v(B) - v(A \cap B) \text{ for all } A, B \in \Sigma$$

Concave utility measures can easily be generated by wrapping a probability measure with a strictly increasing “utility” function  $u$ :

$$v(A) = u(\mu(A))$$

The authors were able to prove that the Dubins and Spanier solution holds in the concave domain too. Similarly they also showed that Hill’s fair border solution is true for concave utilities. The authors do not however mention some benefits of the concave domain. Concave functions are useful because they connect efficiency and fairness. If the valuation function is additive, one of the efficient allocations is to give the whole cake to one agent. Thus efficiency need not induce fairness. If the valuation function is concave however, it is more efficient to give pieces of the cake to different agents since the whole cake is less valuable than the sum of the pieces. Thus an efficient algorithm will also tend to be fairer to all agents. The authors were however not able to determine if the same results would hold if the domain was subadditive rather than concave. (Concave domains are a subset of subadditive domains.) Finally the biggest drawback of the paper is that there is no constructive algorithm offered to actually apportion resources.

A novel approach [48] to resource allocation is to model the preference relations over the set of arcs on a circular cake geometrically. The paper considers the resource to be an infinitely divisible, non-homogeneous and atomless one-dimensional continuum whose end-points are topologically identified. It then proceeds to partition the resource into intervals. It is assumed that the recipients are equipped with preferences over intervals that are additive, continuous and monotonic with respect to interval inclusion. The assumptions about the domain as well as the preference relations are quite similar to the ones in an earlier paper [49]. There are, however, a number of restrictions on the preferences:

- The preferences are smooth, i.e., they have continuously differentiable numerical representations. While this is a simple requirement, it is not trivial to have a preference that is “kink-free” due to the circular shape of the cake.
- Preferences are convex. Again this is a non-trivial condition as convexity depends on the choice of origins.

The paper places emphasis on egalitarian-equivalence solutions rather than envy-free solutions as the former tend to have more pareto-efficient solutions than the latter. Egalitarian-equivalence is a distributional requirement that states that there exists a reference consumption that each agent finds indifferent to its own consumption. Egalitarian-equivalence is however a weaker condition than envy-freeness. Thus all envy-free solutions are egalitarian-equivalent while the converse is not true. Since in many domains the set of efficient solutions are not envy free, egalitarian-equivalence is used as a “compromise” fairness requirement. Other limitations in this paper are that the analysis of preference relations over the union of two or more intervals becomes quite

complex. Trying to model preference relations over the union of at most two intervals requires four-dimensional space. This paper is limited to the domain of one-dimensional resource allocation and does not offer a constructive algorithm for allocating intervals. One of the salient points is that explicit utility functions are not required. Finally the paper also discusses strategy-proofness of their solutions. A rule is *strategy-proof* if no agent ever has an incentive to misrepresent its preferences. Unfortunately for the case of  $n=2$  agents it turns out that any pareto-optimal solution that is strategy-proof is also dictatorial i.e., any one particular agent will always be able to get its most preferred choice of interval irrespective of other agents' choices.

Chambers [50] discusses the various normative properties that land division rules should have based on the principle of utilitarianism. Utilitarianism [51] is a principle in the theory of ethics that prescribes the quantitative maximization of beneficial consequences for a population. The paper studies the circumstances in which rules can satisfy the *division independence* property. If the union of the portions allocated to an agent from various subparcels of a parcel is identical to the portion allocated from the initial parcel, then the rule based on which the allocation is made is said to be division independent. This property may be attractive in case a large piece of resource needs to be broken down into smaller pieces in order to make the allocation problem more tractable. Alternatively if the quantity of a resource keeps increasing over time and the allocation needs to be done over each additional portion incrementally, a rule that is division independent is desirable. The paper goes on to show that any rule that satisfies the above property along with a few other normative properties like independence of infeasible land and efficiency

is a subrule of the weighted utilitarian rule. This is useful since utilitarian social welfare functions have been well studied in literature and can be used to get information about the system.

However the division independence requirement is too constraining and invalidates a lot of other useful properties. For example, division independence is mutually exclusive with the “Strong positive treatment of equals” property. This is discouraging because the above property is a necessary condition for fair solutions to exist. This is a feature of weighted utilitarian rules in general and if the rules are required to be scale invariant as well (i.e., the portions allocated by the rule are independent of the scale of the utility function), then only dictatorial rules are possible. Thus ironically the social welfare approach ends up empowering one individual at the expense of the rest. The other issues are:

- Division independence is meaningful only in the context of additive functions. If the utilities are concave then utility of the union of subparcels will be less than the sum of the utilities of the individual subparcels and, hence, there is no notion of division independence.
- The weighted utilitarian rules require the cardinal comparison of utilities to work. This assumption is not tenable in real world situations.
- The definition of utilitarianism is too vague and cannot be fixed by an objective function.

It is thus clear that division independence is too tight a constraint to be of any significant use in evaluating competing allocation algorithms.

The literature survey shows the multi-faceted nature of the land division problem. Researchers have focused on various aspects of the problem to get a better handle on how it can be resolved. However, each of the approaches also has limitations, either due to the approach taken (measure-theoretic vs. combinatorial) or due to the nature of the domain. This shows that the problem is complex and newer approaches will help.

Chevaleyre et al [52] provide a good overview of the requirements for resource allocation procedures in multiagent systems. Their focus is on solving multiagent resource allocation (MARA) problems and negotiation is part of their framework for solving such problems. Negotiation schemes come under the classification of distributed systems (whereas examples of centralized systems would be combinatorial auctions) and are more scalable as the number of items and agents increases. Negotiation mechanisms are however inherently riskier and the resulting allocations may be not be optimal or efficient. Of particular relevance to this research topic is categorization of resources into types based on the intrinsic property of the resource itself (for example: Is the resource continuous or discrete?) or based on the allocation procedure being used (Is the resource sharable or not among several agents? This will typically depend on the allocation procedure rather than the characteristics of the item itself.). The following criteria can be used to characterize resources (The resource type is indicated in parentheses, ‘R’ indicates that the following is an intrinsic *resource* property while ‘A’ indicates that the property is due to the *allocation* procedure being used):

- Continuous vs. Discrete (R): Continuous resources are assumed to be infinitely divisible. By discretizing continuous resources one can make any methods developed for discrete MARA also applicable to the continuous case. However one may not get the most efficient outcome.
- Divisible vs Indivisible (R)
- Sharable vs Non-sharable (R): A sharable resource is one that can be allocated to a number of agents at the same time. For example, a single picture taken by the satellite can be allocated to several different agents at the same time.
- Static vs Perishable (R): Resources that do not change their properties during a negotiation process are called static. Perishable resources (like flowers) will lose their value when held over an extended period of time.
- Single-unit vs Multi-unit (A): In a multi-unit setting it is possible to have many resources of the same type and to refer to these resources using the same name. In a single-unit setting, on the other hand, every item to be allocated is distinguishable and has a unique name.
- Resources vs Tasks (R): Tasks may be considered to be resources to which agents assign negative utility. Generally any task allocation problem can be reduced to a resource allocation problem. However tasks may involve interdependence on each other which will need to be accounted for during task allocation.

The paper also discusses about the objectives used to enable allocation. This consists of finding an allocation that is optimal with respect to some notion of social welfare. Well known examples of social welfare functions include the utilitarian social welfare that is defined as the sum of individual agent utilities and egalitarian social welfare that tries to

maximize the utility of the current worst off agent. All these various notions of social welfare can be expressed in succinct form by parameterized Collective utility functions (CUF). An example of such parametrized CUF is: the ordered weighted averaging operators as described in the following equation:

$$sw(P) = \sum_i \alpha^{i-1} \cdot u_i(P)$$

Where  $sw$  is the social welfare function that measures the “goodness” of a particular allocation  $P$ ,  $u_i$  is the utility function of agent  $i$ , and  $\alpha$  is a real number between 0 and 1. By varying  $\alpha$ , one can go from the utilitarian CUF ( $\alpha=1$ ) to the leximin CUF ( $\alpha=0$ ). Thus a large family of CUFs can be simulated by simply changing the value of  $\alpha$ .

Finally the paper discusses the complexity issues with respect to the various utility functions and allocation procedures. Negotiation frameworks also need to account for communication complexity. The concept of computational resource is modeled via some formal model of computation. Thus time complexity is the worst-case number of moves made by a deterministic 2-tape Turing machine that correctly classifies the input instances. They summarize the results from earlier literature for quantitative criteria like welfare optimization (An allocation that maximizes  $sw$ ) and welfare improvement (An allocation with greater  $sw$  value than the current allocation). Complexity results are also mentioned for qualitative criteria like Pareto optimality and envy-freeness. The analysis mentioned above holds independently of the regime used to negotiate allocations. Thus analysis of the path convergence of iterative negotiation protocols is a separate but important requirement.



In real world scenarios one has to account for the fact that at each iteration the deals negotiated by agents has to be individually rational, i.e., deals improve the utility of all agents involved. These and other limitations may make it impossible for agents to reach rational deals from the initial starting allocation. Another important consideration is boundedness. Sandholm [53] proves that any sequence of deals that are mutually beneficial will eventually result in an allocation with maximal utilitarian social welfare, provided that agents can use monetary side payments to compensate their trading partners for otherwise disadvantageous deals. Thus any sequence of mutually beneficial deals will converge to a Pareto-optimal allocation.

### ***2.2.3. Fair Division Schemes for Cake Cutting***

#### *2.2.3.1. Background*

The history of fair division is old and interesting. Solomon's proposed solution of dividing a baby, disputed as their son by two women, was probably the earliest known mention of a fair division solution in an informal sense. The problem however has arisen repeatedly even in modern times. Efforts to negotiate settlements have proposed for many such situations. Game theory has been used extensively to model the problem and suggest solutions to rational agents. Real world examples for whom solutions using game theory have been proposed [54] include the camp David accords between Israel and Egypt mediated by President Jimmy Carter. The Cuban missile crisis, which was played out as a game of chicken, is another example. Vote trading occurs in legislatures and company

shareholder meetings, where voters either transfer their voting rights to others for monetary gain or trading across issues. Thus shareholders of group A (say) may vote for removing the CEO (an issue for group B), if voters of group B agree to vote for increasing the dividend (an issue for group A). The conflict between Solidarity, an independent trade union and the Polish communist party, where each side used threats and counter threats to get desirable results is a real world example which has been analyzed from game-theoretic point of view to understand and propose solutions.

In all these problems the attempt is to evaluate the true utility of the desirables to both parties and then come up with a deal that satisfies both of them. One must also make the distinction between arbitration and mediation [55] at this point. Fair division procedures try to make do without an arbitrator, who is authorized to make arbitrary but hopefully fair judgments. Rather a mediator can be used in fair division procedures who elaborates and facilitates the decisions, but cannot pass decisions himself. Fair division depends on the parties coming to an agreement to themselves without external agencies imposing deals on them.

#### *2.2.3.2. Proportional Fair Division*

Gale [56] was the first to discuss the properties of the utility functions that agents could have in order that typical cake cutting algorithms can apply. Specifically the divide and choose solutions requires that the individual preferences are weakly additive. Barbanel and Taylor [57] study this issue further. They introduce an Archimedean property and

show that a preference relation satisfying Gale's conditions is induced by finitely additive measure if and only if it satisfies this Archimedean property. These utility functions set the stage for the domain in which cake cutting algorithms are generally applicable.

Young [58] suggests what he calls the "divider's advantage." It is based on the assumption that the agents' utility functions are common knowledge and everyone acts rationally based on this knowledge. If this is true then the divider is in a much better position than the chooser to exploit the differences in the preferences. If the divider has no knowledge of the chooser's preferences then he will divide the cake into two equal halves based on his valuation. However if he knows what his opponent prefers, he can cut the cake so that it is equal in his opponents valuation. Thus the divider can obtain a larger piece for himself. Young proposes a number of game theoretic solutions to neutralize the divider's advantage.

Crawford [59] describes in detail effects of how a divider's advantage arises in various games. While divide and choose is fair and envy-free, it need not be pareto optimal. A divider can then suggest a solution that maximizes overall efficiency, but almost all the extra utility is accrued to itself. In order to induce the chooser to select a piece (thereby getting the other piece that it views as more valuable), it may add an insignificant amount to that piece thereby making the chooser prefer one piece over the other. In order to neutralize the divider's advantage, Crawford [60] suggests a new fair division device, called the "equal-division divide-and-choose" method (EDDC). EDDC can be administered by the agents themselves and hence do not require an arbitrator to intervene.

EDDC has several desirable properties in a two person pure trade economy, but not all these advantages scale up to the general case of  $n$  agents. Each agent is assumed to seek the most desirable bundle and his preferences can be represented by a strongly monotonic utility function. It is also safely assumed that if a chooser is indifferent to the allocations offered to him, the divider can induce him to pick one piece of the divider's choice. One method to select a divider is to choose one randomly. This is a fair solution, but may lead to "role-envy." Agent A is said to role-envy agent B, if A knows that any agent can get a better utility by playing the role of divider, rather than chooser. One way to remove this asymmetry is to auction off the role of the divider. This can be done implicitly (without money), by letting each agent submit a bid, which is a real number  $\lambda$ , that represents the fraction of a common bundle which the bidding agent is willing to let go. This bidding can balance the advantage of being divider, as the choosers will get equal portions of the fraction that the divider is willing to pay.

Demange [61] further improves upon Crawford's procedure. The drawback in Crawford's procedure is that outside of an equilibrium, an  $n$ -tuple of bids can very well result in a non-feasible allocation of resources i.e., some agents may be assigned a negative amount of goods. Although this procedure is more complicated than Crawford's, it avoids infeasibility, provided it is possible to compute the equilibria in the first place. While non-feasible solutions may be avoidable, the resulting solution may not be egalitarian equivalent anymore. A possible drawback of these papers is the assumption of common knowledge among agents.

Austin [36] proposed a moving knife version of the divide and choose procedure. Since divide and choose can give greater utility to the divider in case of complete information, the process of choosing a divider may itself fall into dispute. A clever resolution of this is by using two moving knives. The cake is assumed to be rectangular in shape (technically any arbitrary shape should be fine as utility of a portion of some length will depend on the icing as well as the shape, but humans might find such evaluation quite complicated). A knife begins moving from the left edge of the cake. As soon as one agent thinks this is about half the cake, it shouts “stop!” The knife is held at this position and a second knife is introduced at the left edge. Now agent A (the one that yelled stop), will hold both knives parallel to the left edge and begin moving them towards the right. In doing so, it will ensure that the piece enclosed by the knives is exactly half by his valuation. At any point agent B can yell “stop!” This will happen because agent B considers the piece between the knives is exactly half in measure. It can then choose either the portion between the knives or the portion outside it. Thus both pieces are exactly half in either agent’s valuation. It is to be noted that allowing agent A to choose a piece after agent B has yelled stop will not be in equilibrium. Also it is guaranteed that there will be a position of the knives as they move from left to right where agent B will think the portion between the knives is exactly half.

Austin’s procedure is not free of problems however. Agent A in this case can be at a disadvantage over agent B. Since agent A yelled stop first, clearly the left portion of the cake was smaller than the right portion in agent B’s eyes. Since it is guaranteed (rules-wise and strategy-wise) that the knives will demarcate the right portion of the cake at the

end of step 2, which agent B considers to be bigger, agent B may never yell stop till the end and simply choose the portion to the right, coming up with a better than 50-50 division. Thus while Austin's procedure can locate a 50-50 split of the cake in both agents' eyes, it cannot induce the agents to choose it.

While Austin's result is constructive, existential results exist for  $n > 2$  players. They go well beyond what Austin's procedure provides. Stromquist and Woodall [62] give the existence result for such a scenario. The agents' utilities are assumed to be non-atomic probability measures. Given that the agents need to agree that same piece is a fraction  $\alpha$  of the entire cake, they show that there exists a subset  $K$  of the entire cake that all agents will commonly agree is of value  $\alpha$  where  $0 < \alpha < 1$ .  $K$  can be at most a union of  $n$  intervals. The results are shown for many rational and irrational values of  $\alpha$ . But for some rational values of  $\alpha$ , the authors are unable to determine the minimal number of cuts that will suffice.

Banach and Knaster discovered the procedure for extending the classical divide and choose solution from two to  $n$  agents. This procedure is elaborated by Steinhaus [63] in his paper on fair division. It consists of arranging the agents from 1 to  $n$  with agent 1 cutting of a piece of the cake which it considers to be  $1/n$  (this not part of the rule, but doing this is rational for agent 1). The piece is passed down the line of agents, with each agent having the right to trim off a portion if it feels the piece is worth more than  $1/n$ . The rules however state that the last person to trim the cake will have to accept it if the other agents reject it. The agent that accepts this piece (either voluntarily or because it was the

last trimmer) is out of the game. The remaining  $n-1$  agents now start with the procedure all over again, till finally there are only two agents left to be allotted portions. These agents can use the standard divide and choose to distribute the remainder of the cake among themselves. This procedure is able to allocate fair portions of the cake to all agents, but one loses the envy-freeness available in divide and choose. Also the running time for this algorithm is long,  $O(n(n+1)/2)$ . This procedure is applicable to continuously divisible objects like a cake.

Steinhaus also elaborates on Knaster's procedure for  $n$  agents attempting to divide  $k$  indivisible object among themselves. Here a mediator agent will make note of each agent's estimations of the objects. Then the object will be assigned to the highest bidder. It is quite possible that some agents may be assigned multiple objects while others may not receive anything at all. In order to remedy this, each agent which has received more than  $1/n$  of its share is made to contribute the excess utility (in money units) to a common fund. The agents, who still have negative utility, can be compensated from this fund. Thus in order to receive its fair share, agents may be allotted a mix of goods and money. While this scheme is certainly useful, it requires that a pool of extra cash be available to the agents, which will be used as the divisible "object" to give everyone a fair share.

Brams and Kaplan [64] also look at the issues of apportionment of indivisible resources among a set of competing entities. They take up the specific example dividing ten cabinet ministries (the indivisible resources) among various parties (the competing entities) in the coalition government of Northern Ireland. It is assumed that each party covets as many

cabinet ministries it can get and that there is a preference ordering for the ministries. Based on the divisor method of apportionment, they show that rational parties can make choices that are non-Pareto-optimal. They suggest a mechanism combining sequential choices with a structured form of trading that improves efficiency. This mechanism is not scalable to more than two agents however. The authors propose the Gap procedure [65] to divide indivisible goods among  $n$  agents. The agent bids determine the prices for the goods. The procedure ensures the following properties:

- (Nonnegative) prices that do not exceed the bid of the winning agent.
- Pareto optimality.
- Monotonicity: Higher bid value will result in higher chances of obtaining the good.
- Truthful bidding
- Partial independence of prices from the bidding amounts.

Issues like presence of envy need to be resolved however in order to make the procedure the more appealing to solving real-world problems.

Schneider and Kramer [66] have collected empirical data on the relative benefits and drawbacks of three well known procedures, namely: Proportional Knaster, Adjusted Knaster and divide-and-choose. They find that as long as participants are forced to adhere to bargaining protocols, Proportional Knaster and Adjusted Knaster give better solutions than the standard divide-and-choose. However if deviating behavior of the participants is tolerated, then divide-and-choose performs better. This seems to happen because divide-and-choose derives its rules from a game-theoretic basis and the assumption of non-cooperation is built in. The authors mention the inadequacy of current fair division



procedures because of the assumption that the participants have no common past or future.

Kuhn [67] expanded greatly the possibility of Steinhaus' procedure for three agents. While Brams and Taylor have called it the lone divider procedure (to signify that one agent divides, while the rest choose), this may be misleading. As per Steinhaus' proposal (for  $n=3$ ), one agent will become the divider and will cut the cake in three pieces (equal in its eyes). Then agent 2 and agent 3 will evaluate the pieces. If they think two or more pieces are acceptable then one goes to agent 2 and the other goes to agent 3, while the last piece is allotted to agent 1. If the choosers (agent 2 and agent 3) think that only one piece is acceptable (and it is the same piece), then they will be two pieces that both find unacceptable one of which is handed over to agent 1, while the remainder is combined and divided among agent 2 and 3 using divide and choose. If the choosers think at most one piece is acceptable, but they are two different pieces, then they will each take those pieces and give the remainder to agent 1. Thus combinatorially, all possibilities have been accounted for. Note however that there is redivision involved and other agents may wield the knife at later stages than the purported divider (agent 1).

Steinhaus does not mention if the procedures apply for  $n>3$  players. But Kuhn provides an existential proof that this is possible. They use the combinatorial theorem by Frobenius and König to prove that such a procedure can find allocations for arbitrary  $n$  agents. A rigorous framework is setup for generalizing it to  $n$  agents. First, the divider is chosen among the agents randomly. Then, the notion of "fair restriction" is defined. Fair

restriction is a reduction of the division problem to a possibly smaller set of  $m$  players ( $1 < m \leq n$ ) by removing  $n-m$  parts of the cake, none of which is acceptable to the  $m$  players remaining. The framework reasonably assumes:

- Each player will find at least one of the pieces to be acceptable in the worst case.
- The divider should find all the pieces to be acceptable. (Since it performed the division)

The framework setup then cleanly plugs into the properties of the Frobenius-König theorem due to which either there is a complete assignment of the cake or the agents unsatisfied with some portions of the cake have the remainder re-divided among themselves in the next iteration. Since at least one agent comes out satisfied, in each iteration, the allocation procedure stops in a finite number of steps. Kuhn however does not explicitly layout the algorithm to assign portions to agents. Also proportionality is assumed to be the only criterion. This however is no guarantee that there will be no dispute among two agents (in a 3 agent division), both of whom find at least two pieces acceptable, since they may both find one piece to be larger than the other and fight over the claim to that piece.

Kuhn also elaborates on Knaster's procedure for the division of indivisible goods using side payments. Knaster's procedure involves agents submitting valuations of individual objects in a bundle to a mediator. Kuhn shows that Knaster's rule can be discovered through linear programming. An assignment matrix  $X = (x_{ij})$  is defined with  $x_{ij} = 1$  if object  $i$  is assigned to agent  $j$  and  $x_{ij} = 0$  otherwise.  $y_j$  denotes the side payment to or from

agent  $j$  and  $v_{ij}$  is agent  $j$ 's valuation of object  $i$ . The linear programming problem is set up as follows:

$$\sum_i v_{ij} + y_j - v_j \geq z \quad (j = 1, \dots, n)$$

$$\sum_j x_{ij} = 1 \quad (j = 1, \dots, n)$$

$$\sum_j y_j = 0$$

The constraints are thus set up and then one tries to maximize the surplus  $z$  over the fair share which can be given to all players, using side payments  $y_j$  which sum to zero. As useful as these results are, there are certain drawback to these approaches as well. Using a simple example Kuhn shows that the system allows choosers to benefit over dividers. This is because dividers will get exactly  $1/n$  of their share, but choosers get to divide larger sized portions among themselves. Knaster's proposal similarly also has drawbacks. If agents have a good idea of the opponents' valuations, they can maximize the utility they get by overbidding just enough to trump everybody else's bid (if they wanted to win the item) and underbidding just enough to lose to the highest bidder (if they wanted to lose the item). By lying in this manner the agent will be able to increase his profits. Thus Knaster's procedure does not induce truthfulness among agents.

Dubins and Spanier [43] were the first to mention of the moving knife procedure. The procedure essentially calls for a knife hovering over a rectangular piece of cake and slowly begins to move in a straight line parallel to one of the cake's sides. The knife is held by a mediator, while the  $n$  agents evaluate the size of the portion to the left of the

knife. Any agent that shouts “cut!” will get the piece traversed by the knife till that point. The agent is then removed from further consideration of the cake. The remaining players continue to play the same procedure to get their respective allocations. The allocation is finished with  $n-1$  cuts. This is a very elegant solution to the  $n$  agent cake cutting problem which is frugal in the number of cuts involved. It is the only known algorithm that can accomplish allocation with the fewest number of cuts possible. Actually the authors describe the procedure as an exposition of Knaster’s solution for trimming which extended divide and choose to  $n$  agents. But there is a clear difference between their procedure and Knaster’s trimming procedure. In Knaster’s procedure players do not need to make continuous choices over the portion of the cake. Knaster’s procedure is discrete in nature and is applied in steps. Thus the Dubins-Spanier moving knife procedure is distinct from Knaster’s procedure. It uses continuous type procedure to its benefit by achieving allocation with very few cuts.

But the same procedure also causes problems. The moving knife procedure roughly corresponds to the Dutch auction where the price of the auctioned good is continually lowered, till finally one agent accepts the bid. It therefore suffers from problems similar to what auctions do i.e., it is vulnerable to corruption by the auctioneer (mediator) who may be biased towards or against one or more agents. The moving knife solution is unverifiable and cannot be implemented algorithmically in the computer. In case of a computerized version of the procedure, synchronization of agent calls becomes a very important requirement. The calls cannot be serialized (since the agents do not have a common measure function) and it has to be explicitly clear to all agents as to who made

the first call. While being proportional, the procedure is not envy-free. An agent, who receives a portion initially and is out of the race, may feel that a later agent got a larger share than its own even if it got  $1/n$  of the whole cake. The earlier agents may thus feel envious of later agents in this procedure. Only the last two agents feel no envy, as the remainder of the cake can be divided using divide and choose rather than continuing with the moving knife. Thus any surplus they get can be evenly divided among themselves.

Dubins and Spanier also discuss the issue of partitioning a cake into  $n$  pieces such that each of the  $n$  agents assigns the same value to all pieces. They take inspiration from a specialized version of the problem called the “Ham Sandwich Problem” (attributable to Steinhaus [68]). The name is inspired by the question of whether a ham sandwich composed of ham, butter and bread can be cut by a knife into two halves such that each ingredient in the sandwich is halved. The general problem involves simultaneously cutting in half each of the  $n$  bodies in Euclidean  $n$ -space by a hyperplane. This can be transformed into the cake cutting problem. Given  $n$  finite measures in euclidean  $n$ -dimensional space such that each measure vanishes on any set whose volume is zero, is there a hyperplane bisecting the space with respect to each measure? First they present a proof of the Ham Sandwich Problem by making use of properties of the Borsuk-Ulam theorem. This is then used to prove the General Bisection theorem and finally applied to make  $n$  divisions of the cake, such all agents give the same value to all portions. This, however, is an existence result only. No algorithms have been provided to compute such a solution.

Fink [69] proposed another mechanism by which a cake can be distributed among  $n$  agents. It consists of applying the divide and choose procedure first to two agents. Now agent 3 is introduced and agent 1 and agent 2 divide the cake their portions into three equal pieces each. Agent 3 then gets to pick any one of those pieces from agent 1 and agent 2. Next, if agent 4 comes along, the division procedure is now repeated, but now by all three agents (1, 2 & 3) do so. This procedure terminates when all  $n$  agents have been satisfied. One advantage of this approach is that the division process works for uncertain number of agents. In a dynamically changing situation, if the final number of agents is unknown, Fink's procedure can continue to allocate incrementally. Again, this process is proportional but not envy free. Another issue is that, the agents who come in earlier in the process may have to do more work than the agents who come in later. Also since earlier agents will be the dividers and the later agents will be the choosers, the later agents will stand to get more benefit than the earlier. This is equivalent to the problem in the two-agent divide-and-choose case, where the chooser can possibly get a larger than proportional share while the divider gets exactly 50% of the whole. The number of cuts that Fink's procedure entails is also large ( $O(n(n-1)/2)$ ) and the largest burden is placed on the first divider of the cake. The desirable aspect is that this procedure can be initiated without knowing how many players will eventually be involved and is able to scale up dynamically to changing situations.

Woodall [70] uses Fink's procedure to come up with an even better procedure for fair cake division. It was well known that unless agents' utility functions are identical, the differences in their evaluation of a particular piece can be exploited to give each agent

more than a fair share i.e., more than  $1/n$ . Such a scheme would be called super fair. Note that this does not mean that the division is envy-free, rather it is more than what the agent thinks is proportional. Thus there exist cases where envy will occur despite getting more than  $1/n$ . Only in the  $n=2$  case, fairness and super fairness implies envy-freeness.

Woodall describes an algorithmic way to support the existential result. For  $n=2$  agents, if agents 1 and 2 value a piece of cake to be  $\alpha$  and  $\beta$  respectively, choose a rational number  $p/q$  such that  $\alpha > p/q > \beta$ . Now make agent 1 divide this piece into  $p$  equal pieces while agent 2 divides the remainder into  $q-p$  equal pieces. Agent 1 chooses one of  $q-p$  pieces (say  $x_2$ ) to allot to agent 2 that it considers to be less than  $1/q$ . Agent 2, similarly, chooses one of  $p$  pieces (say  $x_1$ ) to give to agent 1. Since these pieces are from the parts of the cake that each agent values more, both have the perception that they have an irrevocable advantage over the other. One now removes  $x_1$  and  $x_2$  from consideration and use the standard divide and choose procedure over the rest of the cake. Thus each agent gets more than  $1/n$  of the cake.

Woodall then uses Fink's procedure to divide the cake among  $n$  agents. At each stage every agent with a cake will make  $k$  equal portions of the cake (if the  $k^{\text{th}}$  agent is being considered), but using Woodall's procedure cuts the  $k$  portions into sufficiently  $q$  small sections so that there are  $qk$  pieces and the portion remaining after  $k^{\text{th}}$  agent takes out  $q$  pieces is still greater than  $1/k$ . Agent  $k$  will also similarly think that it has more than  $1/k$  as it gets to pick the  $q$  largest pieces from each of the  $k-1$  agents. In terms of the number of cuts however this procedure does even worse than Fink's procedure. While Fink's

procedure required  $k$  cuts by each agent at the  $k^{\text{th}}$  stage, Woodall requires  $kq$  cuts in the  $k^{\text{th}}$  stage, where  $q$  can be arbitrarily large. The evaluation of  $q$  requires an intimate knowledge of the opposing agent's utility function. This can be an unreasonable assumption in a heterogeneous multiagent system.

Berliant, Dunz and Thomson [71], discuss extensively on the relationships between the various equity criteria used to evaluate cake cutting procedures. Brams and Taylor [14] [72] have asserted that in general any two of three criteria are achievable. The criteria are:

- discreteness (being algorithmic)
- proportionality
- efficiency

Berliant, Dunz and Thomson discuss utility functions which give proportionality, envy-freeness and efficiency. Their results are, however, existential in nature. Among the reasonable assumptions made are that utility functions are nonatomic and additive. The good is heterogeneous, with each agent having a preference relation over various parts of the good. The paper focuses on getting results that are envy-free and efficient (proportionality is automatically satisfied as envy-freeness implies proportionality). The assumptions on the nature of agent utilities are gradually relaxed, from additive to being sub-additive. The sub-additive case is attractive to economists as this connects with the notion of decreasing marginal utility, which is how most people evaluate objects in the real world. But they also mention that allowing arbitrary preferences over objects is



unproductive to getting good quality solutions. The attempt is to relax the assumptions as much as possible.

They extend existing results to show existence of:

- $\alpha$ -fair efficient allocations:  $\alpha$ -fair is a notion of fairness where each agent perceives the utility of the portion allocated to him as being  $\alpha^{\text{th}}$  part of getting the whole good
- Envy-free allocations
- Efficient envy-free allocations
- Group envy-free allocations. An allocation is group envy-free if for any two same-sized groups of agents no partition of the portions allocated to a group satisfies the other group.
- Allocations that give out nicely shaped portions with the above properties. This is a desirable feature in case the resources are like land, where long wiry shaped portions are less preferred over compact shaped allocations.
- Egalitarian-equivalent efficient solutions.

The group envy-free criterion is particularly useful because by tuning the parameters one can go from being envy-free to being efficient. Consider two groups of agents  $C_1$  and  $C_2$  containing the same number of agents. If an allocation is known to be group envy-free, then setting  $C_1$  and  $C_2$  to be of size one i.e., each group contains one agent, one gets the following statement:

Agent  $x_{C_1}$  does not envy any reallocation of portions allocated to group  $C_2$ , but this group contains only one agent. So agent  $x_{C_1}$  does not envy agent  $x_{C_2}$ . This argument can be applied to all agents and hence the whole allocation is envy free.

Now consider the two groups whose size has grown to cover all agents i.e., all agents are present in both the groups. If the allocation is known to be group envy-free then no reallocation of portions allocated to group  $C_2$  will satisfy any one in  $C_1$ . This means that the existing allocation is efficient. Thus the group envy free criterion is very useful in connecting envy-freeness with efficiency.

Egalitarian-equivalent solutions are desirable in allocations. Generally existence of egalitarian-equivalent and efficient partitions requires less structure and weaker assumptions than those for envy-free and efficient partitions. The authors however found no systematic set-theoretic relation between egalitarian-equivalent and envy-free allocations. The authors also discuss the availability of envy-free and efficient allocations in the domain of time scheduling problems. Scheduling time for use of common facilities slightly differs from the standard cake cutting domain. While in cake cutting one can divide and combine different portions of the cake infinitely many times, time scheduling is only infinitely divisible. Due to real world constraints an agent would prefer a large continuous time slot rather than many small time slots all adding up to the same duration. Therefore the authors reasonably assume that the utility of a union of time intervals allotted to an agent is the same as the largest utility of the maximal intervals it contains. They are able to show a strong result that any envy-free allocation is necessarily efficient and in fact group envy-free. While this is very encouraging, it remains to be seen if these results are monotonic with respect to the amount to be divided and to the number of agents that will participate in the allocation. The authors also do not provide any

procedures that can find these desirable solutions algorithmically. The relationship between fairness and envy-freeness is discussed in greater detail in Section 3.3.6.

#### 2.2.3.3. *Maxmin Fairness*

The maxmin fairness criterion is part of the social justice literature. Social justice is based on the idea of a society which gives individuals and groups fair treatment and a just share of the benefits of society. Thus the purpose of maxmin fairness is to maximize the minimum allocated to each agent whose demand has not been fully satisfied. The issue of maxmin fair allocations of a resource has been studied by researchers in computer science, mathematics and social justice.

Bezakova and Dani [73] apply the maxmin fairness criterion for the allocation of indivisible goods among  $n$  agents. Using a randomized cut-and-choose procedure they show that a  $1/2$ -approximation of the optimum is achievable. Another algorithm that calculates fractional optimum is presented, but it has an additive error bounded by the value of the largest item. The algorithms, however, are incomparable because they outperform each other in various situations. Asadpour and Saberi [74] extend the work to present a nontrivial approximation algorithm with a much improved approximation ratio.

Kumar and Kleinberg [75] look at issues of maxmin fairness for solving real world problems of bandwidth allocation, scheduling and facility location of warehouses. They

show that while approximate solutions are appealing (because the problems are NP-complete), it is not guaranteed that even an approximate solution exist in the first place.

Brams and King [76] study the interactions of social justice criteria like maxmin fairness and Borda maxmin fairness with traditional cake-cutting criteria like envy-freeness and efficiency. They show that the two maxmin criteria are in conflict with ensuring envy-freeness. Based on computer simulations they show that although envy-freeness cannot be guaranteed, Borda maxmin allocations have a greater chance of being envy-free as compared to simple maxmin allocations.

Dutil [77] discusses the connections between social choice and equity theories. Social choice deals with the idea of aggregating information about individual preferences to form a collective preference, which can then be used as a metric to measure the extent of “social good” created. Equity theory deals with the notion of distributive justice. Thus there exists a good deal of synergy between the two objectives. The discussion in this paper provides context for research in fair division such as the criteria that need to be considered for defining “fairness.”

Gersbach [78] examines the division of resources among individuals using voting rules. Rather than use either simple majority rules on one hand or unanimity rules on the other, a hybrid of the two is proposed. In order to mitigate the unevenness of the distributions, the “flexible majority rule” requires that the majority required for adopting a proposal increase with the amount of inequality the proposed distribution creates. This rule

therefore induces efficient and even distribution of resources. However in cases where individuals differ strongly in their preferences or multiple commodities need to be distributed, it is unclear how flexible majority rules are to be applied.

#### 2.2.3.4. *Envy-Free Fair Division*

Attaining envy-freeness in resource allocations has been the holy grail of research in fairness criteria. In case of  $n=2$  players, the divide and choose procedure is trivially envy-free. If each player thinks it got a fair share of the cake (i.e., not less than half the cake) then it would also think the share is envy-free (i.e., its piece is at least as large as the other agent's). This serendipitous matching where fairness implies envy-freeness and vice-versa breaks down once  $n \geq 3$  players are in contention for resources. Envy-free criteria are discussed in greater detail in Section 3.3.2.3

Brams and Taylor have studied the issue of obtaining procedures for  $n \geq 3$  intensively. In general the cases have to be dealt with individually and come with caveats. Steinhaus [63] provided proof for the existence of envy-free allocations for  $n \geq 3$ , but did not suggest a constructive procedure to get to such allocations. Brams mentions the Selfridge-Conway procedure to get envy-free allocations for  $n=3$  agents. It can be completed in a maximum of 5 cuts. The approach taken by most envy-free procedures involves players trimming down pieces created by other players. The trimmings are generally set aside and become inaccessible to all players. This reduces the efficiency of the allocation procedure. Thus many procedures trade efficiency for envy-freeness. The Selfridge-

Conway procedure is ingenious because it manages to allocate the trimmings as well. Thus there is no wastage of resources in the allocation process.

Brams et al [79] have proposed two procedures which are not only envy-free, they are equitable and frugal (in terms of number of cuts) for the agents involved. The surplus procedure (SP) is truth inducing but is only applicable to two agents at a time. The equitable procedure (EP), on the other hand, can be applied to any  $n$  number of agents but is not envy-free. They discuss the applicability of SP and EP to land division.

Stromquist [35] proposes a moving-knife (continuous) version for 3 agent cake division. The procedure involves all 3 agents in addition to the referee holding the knife parallel to the mediator's knife, moving from the left edge of the cake to the right. The agent knives hover over the piece of the cake that is to the right of the referee's knife and each agent positions it so that the right portion of the cake is divided evenly into two halves. Based on the rules of the procedure, each player is ensured an envy-free allocation. This procedure is envy-free and frugal because it manages to get the requisite allocation with the minimal number of cuts.

A different take on the moving-knife procedure is suggested by Levmore-cook. It involves two knives one vertical (like in traditional moving-knife) and one horizontal. Agent 1 first creates 3 equal portions (in its valuation) of the cake using the vertical knife. If Agents 2 and 3 think different pieces are the largest, an envy-free allocation has been found and the procedure terminates. But if agents 2 and 3 consider the same piece to be

the largest, then agent 1 now uses the horizontal knife in conjunction with the vertical knife to trim the largest piece (these trimmings are added to the other pieces) till one of agents 2 or 3 does not desire this piece anymore. This procedure is also efficient.

The Webb moving-knife procedure describes one more way to get allocation for  $n=3$  agents. To begin with a knife is moved over the cake (from left to right) till one of the agents, (say agent 1) calls cut (because it thinks the piece to the left is worth  $1/3$ ). Then agent 1 gets together with agent 2 and 3(both) use Austin's procedure over the remainder of the cake to create two equal portions. Agent 3 then gets to choose the first piece, followed by agent 2 and then agent 1. This procedure gives an efficient envy-free allocation of the cake. A slightly different way is for agents 2 and 3 to use Austin's procedure and divide the cake into 3 equal portions (in both their valuations). Then let agent 1 choose first. This too results in an envy-free and efficient allocation.

Brams, Taylor and Zwicker [37] propose a 4-agent envy-free allocation procedure. It consists of agents 1 and 2 using Austin's procedure to create 4 equal portions of the cake. Agent 3 trims one of the pieces to create a two-way tie for the largest and allow agent 4 to choose first. This allocation is envy-free, but trimmings created by agent 3 still remain to be allocated. The authors add a follow-up procedure based on the notion of "irrevocable advantage" to divvy up the trimmings as well.

As seen in the discussion so far, the procedures for allocating envy-free portions to 3 or 4 agents required quite a bit of ingenuity. By marrying the different allocation procedures

for  $n=2$  researchers have been able to come up with such procedures. It is therefore easy to imagine the difficulty of getting procedures for general set of  $n$  agents. Brams and Taylor however manage to propose not just one but four different procedures to achieve such allocations.

The first procedure is an adaptation of the traditional moving knife procedure (which can only allocate fair portions). Unlike traditional moving knife, it allows the players who have already been allocated portions to "re-enter" the game in case they find the currently considered portion of the cake to be larger than the one they got. They specify a tolerance,  $\varepsilon$ , where any pieces smaller than  $\varepsilon$  are considered to be negligible by the player. Thus a player may call cut at most  $1/\varepsilon$  times in the worst case before being satisfied with his piece. It is to be noted that this procedure is only approximate envy-free, but none of its real world appeal is lost as in practice  $\varepsilon$  can be made small enough to satisfy every players envy-freeness.

Another approximate envy-free procedure (to begin with, consider only two agents) consists of agent 1 creating  $2n+1$  equal piece of the cake. Agent 2's job is to partition these pieces into two equal halves in its valuation with the proviso that the two halves will differ by atmost 1 piece. Brams and Taylor mention of a procedure that can achieve this. Since the two pieces will differ by 1 piece (and assuming agent 2 takes the one with more pieces), agent 1 will get a portion that is atmost  $1/(2n+1)$  smaller than the other. For large enough  $n$ , this may be small enough to be negligible for agent 1. This procedure is expanded to allow two agents to create an arbitrary number of equi-valued pieces (in both



their eyes) within the limit of the tolerance set. The procedure can be extended to include a third player as follows: In the description above replace the term "agent 1" with the term "agent 1 and 3". Then have agent 2 do its usual part. By inductively adding more such agents one can have a procedure for arbitrary  $n$ .

Another discrete procedure that produces exact envy-freeness (at the expense of efficiency and boundedness) is an adaptation of the Selfridge-Conway procedure. Consider  $n$  agents; agent 1 creates  $2^{(n-2)}+1$  equal portions of the cake then passes them on to agent 2 who then trims at most  $2^{((n-1)-2)}$  pieces to tie for the largest. In general the  $k^{\text{th}}$  player will trim at most  $2^{(k-2)}$  pieces to ensure itself of a piece that is atleast as large as any other piece. However the procedure is less efficient because trimmings that are left over are not allocated to anybody. This can be remedied somewhat by running the procedure again over the trimmings. After a finite number of procedure runs, the trimmings will be small enough for every agent so as to be considered negligible.

Brams and Taylor present a finite version of the above mentioned infinite procedure using the notion of irrevocable advantage. It consists of playing the Irrevocable Advantage subgame that allows an agent (or group of agents) to be envy-free of even if another agent (or group of agents) gets all the trimmings. The agents in the second group then divide the trimmings equally among themselves. The procedure is however quite complicated and in general involves a large number of cuts to be made. Finally Brams and Taylor suggest the notion of a C-procedure (for cut) whereby procedure can create  $n$  portions of the cake with  $n-1$  cuts. If these portions are such that no other C-procedure

yields an allocation that is strictly better than this one, then the procedure is called C-efficient. As it turns out any envy-free C-procedure is C-efficient. Stromquist's three-person moving-knife procedure is C-efficient. In economics theory, envy-freeness has become as important a criterion as pareto optimality for judging the "goodness" of allocations.

Raith and Su [80] describe a procedural approach for implementing envy-free and efficient allocation of indivisible goods among agents. This is elaborated further in Haake et al. [81]. The authors require that the agent utility functions be quasi-linear. Envy is eliminated by compensation of envious players using monetary side-payments. The procedure is similar to the Adjusted Winner (AW) procedure proposed by Brams. Consequently it suffers from the same drawbacks as AW, viz. vulnerability to strategic manipulation.

#### 2.2.3.5. *Redefining Envy-Free*

Arnsperger [82] evaluates how well envy-freeness can stand in as a criterion for distributive justice. Various new approaches to getting envy-free allocations that have come up in literature are mentioned. One point of interest to be noted is that envy-freeness is a purely ordinal criterion of distributive justice. It relies on no interpersonal comparisons of utility whatsoever. There has been considerable debate on what semantic notion envy-freeness stands for. For example in the real world jealousy (or envy) may involve a visible disutility for the person that is feeling envy. But in the multiagent

system one can safely fix the meaning of envy freeness to mean that the agent is convinced that it has received the largest portion of the resource. By such a definition, pareto efficiency(with respect to permutations) is also connected to envy-freeness. An allocation is Pareto efficient with respect to the permutations if it is unanimously preferred to all the allocations where elements of the current allocation are swapped between agents. Envy-freeness is quite useful as an equity criterion in societies where preferences and endowments are heterogeneous. A detailed discussion of various alternatives to envy-freeness appears in Section 3.3.2.1

Arnsperger [82] also discusses the issue of whether envy-freeness is informationally economical because in general checking an allocation for envy-freeness requires making  $N(N-1)$  comparisons, which can be expensive once  $N$  grows large. The paper provides useful results showing the existence of envy-free allocations for various situations. An interesting extension of envy-freeness is the notion of opportunity envy-freeness. Opportunity envy-freeness is when no agent envies any other agents choice set (A choice set being all the possible choices of portions an agent would accept). It is to be noted that criterion is actually stronger than plain envy-freeness. Thus opportunity envy-freeness implies envy-freeness but the converse is not necessarily true. The author checks to see if envy-freeness is compatible with the axioms of population monotonicity and resource monotonicity. These axioms require that when a characteristic of the economy changes, all agents should be affected in the same direction. Thus the losses due to population increases should be borne by all members of society and gains due to resource increases should benefit everyone in the society. Unfortunately as it turns out solutions that are

both envy-free and efficient fail to satisfy either of the monotonicity axioms. Due to the difficulty of achieving envy-free allocations, the paper suggests two alternative approaches.

The first is trying for envy-minimizing allocations. The goal in this case is to minimize the envy in society. The paper sets up a metric for measuring the envy of an agent as the fraction of the opposing agent's portion that would equal its own portion. This measure (appropriately modified) is summed up over all agents and the goal is to find an allocation that will minimize the total envy.

The second approach is to weaken the envy-freeness condition in order to arrive at new, though closely related criteria for which allocations exist. Such a suitable criterion would be "Absence of dominated agents," i.e., rather than have agent 1 alone judge his situation compared to agent 2, all agents get to voice their opinion on the matter. Thus no agent would be dominated (in a appropriately defined sense) by any other agent. Thus the "rigidity" of the envy-freeness requirement is avoided by "engineering" approaches and criteria that closely compare to the ideal of envy-freeness in spirit, but are still flexible enough to allow feasible solutions.

Fishburn and Sarin [83] do a detailed analysis of the various indices that can be used to measure fairness of allocations. They determine these indices based on what parameters of a population need to be emphasized upon. One dimension along which indices can be created is the level of aggregation that needs to be taken into consideration. For example:

Are fairness issues being examined at the level of individual agents, groups of agents or the entire population taken together? The paper considers each feasible allocation to be a "profile" of risks and benefits. An agent is supposed to be capable of ordering benefit-risk pairs to indicate its preferences. Their framework only requires this ordinal information from each agent. Utility functions are not required for finding feasible (and optimal) allocations. However it is possible to create utility functions that can generate these orderings on benefit-risk pairs. These "shell" utility functions enable easier analysis of the indices. Another dimension which affects quality of allocation is the characterization of the benefit-risk profile, i.e., Are individual profiles to be considered or should one consider the joint probability distribution of benefit-risk profiles (in case of uncertain situations) or further still simply compare the marginal probabilities of the agents to deduce envy? Each of these aspects has its advantages and drawbacks.

For example, one can state that one agent envies another if it prefers the other's marginal probability profile to its own. Given such a description of envy it can be shown that free market mechanisms produce a fair solution from the marginal perspective. However the joint distribution of benefits and risks might reveal the presence of considerable envy in some groups. The authors set up a framework and describe three possible indices that can be used to measure envy. The first measure of fairness is the average number of others that an individual agent envies. Consider a population of  $n$  agents, if  $x$  is the vector of profiles (allocations), then use  $e_{ij}[x]$  to denote the envy of agent  $i$  for the portion of agent  $j$ .  $e_{ij}[x]$  is 1 if agent  $i$  envies agent  $j$  and is zero otherwise. Then:

$$e_i[x] = \sum_{j \neq i} e_{ij}[x]$$

is the number of people that  $i$  envies in  $x$ . Then index  $m_1$  is a measure of fairness defined as:

$$m_1[x] = \sum_i e_i[x] / n$$

Thus minimizing  $m_1$  to maximize fairness can be used as a criterion for choosing one profile over another. Another useful way to measure fairness of a profile is to find the number of agents which envy more than half the number of other agents based on portions allocated to them. Thus an allocation is “median fair” if an agent does not envy more than half the other agents in the population based on their allocated portions. This can be defined as:

$$m_2[x] = \{i : e_i[x] > (n-1)/2\}$$

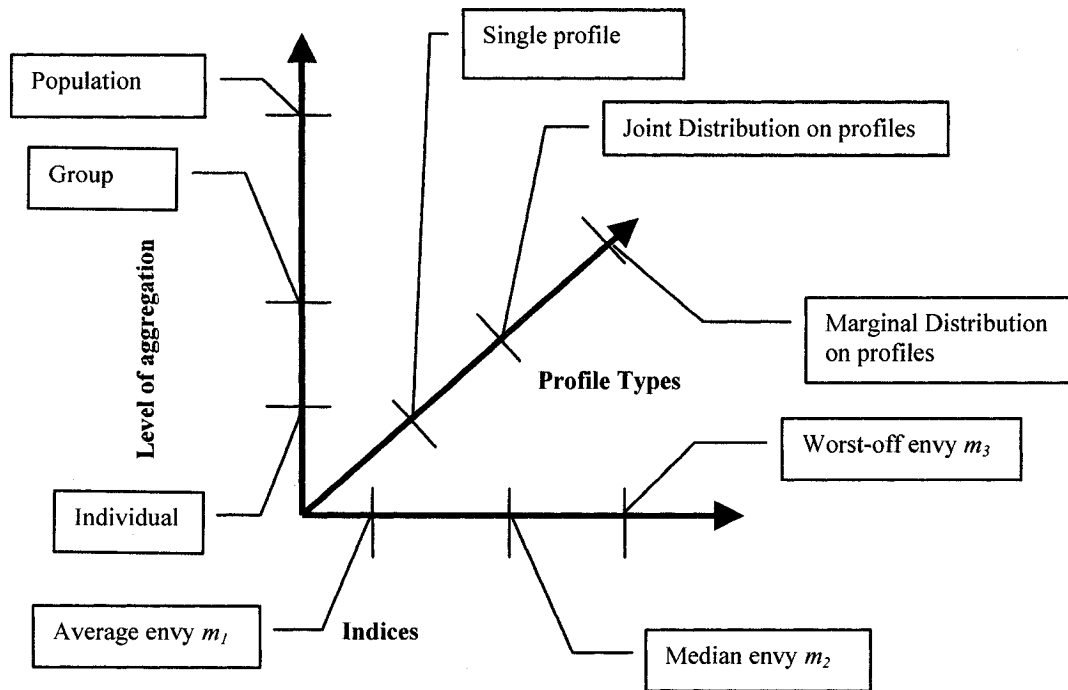
This is a cruder measure than  $m_1$  but is easier to assess in many circumstances. The third measure is based on Rawls’ idea of making the worst-off person as well off as possible. Thus:

$$m_3[x] = \max_i e_i[x]$$

It should be noted that the above three measures are defined in the case of simple profile analysis at the individual agent level. The authors define and analyze these measures for the complete space of profile types and aggregations. Figure 7 shows the various cases that have been accounted for.

The authors thus provide a way of choosing the index that best suits a specific situation. The paper however does not comment on the conditions for envy-free solutions to exist

nor any constructive methods to reach the envy-free allocations. However it does provide indices that relax the connotation of envy-freeness by various degrees so that the space of feasible solutions is enlarged.



**Figure 7.** Different ways of measuring envy-freeness

# Chapter 3

## Evaluating Negotiation Protocols and Procedures

The literature survey shows the different approaches taken by researchers from various fields. Researchers in MAS tend to focus on how to design mechanisms and protocols that discourage manipulation by agents and enable stable interactions. Economists, on the other hand, have focused on finding mechanisms that allow envy-free allocations to exist. Their focus is less on procedures that enable us to find such an allocation. So their arguments are more existential rather than constructive in nature. The third group consists of researchers from mathematics who have also contributed greatly to exposing the various issues in resource allocation among autonomous agents as well as proposing constructive solutions to attain such allocations.

The mathematical literature discussing these problems has existed from at least as early as 1948, when Steinhaus [63] explicitly discussed the problem of dividing a cake among  $n$  agents. The paradigm for discussing negotiation protocols/allocation procedures is different depending on the background of the researcher involved. Mathematicians primarily modeled the resource as a cake or a pizza that had to be divided among various people with differing evaluations over various portions of the (edible) resource. Why the cake was a preferred paradigm was never explained. But, based on the procedures that



mathematicians have suggested it is easy to see that two properties were assumed to be true for the resource being divided: the mathematicians implicitly assumed the resources were *infinitely divisible* and *recombinable*. Indivisible resources, such as a large estate being divided among heirs, have been looked into as well, but the solutions involve selling off the house (to one of the heirs or to an external party) in lieu for “liquid” money, which is then used as a divisible (and recombinable) resource to satisfy the contending parties. Fairness (where an agent in a group of size  $n$  believes it received at least  $1/n$  of the resource), envy-freeness (where an agent believes it received a portion larger than everyone else’s), and efficiency of allocation procedures, are of major concern. Boundedness (i.e., complexity of the allocation procedure) and running times are discussed, but they are not the major concern for mathematicians.

Economists on the other hand have focused mainly on existence conditions for envy-free allocations. Their treatment is set-theoretic and they put forth axioms and properties of the domain for existence of envy-free solutions. Since envy-freeness is generally difficult to achieve, there has been considerable interest in putting forth alternative criteria that keep the essence of envy-freeness but relax the constraint so that feasible solutions exist. An example of such an alternative is egalitarianism which will be explained later in this chapter.

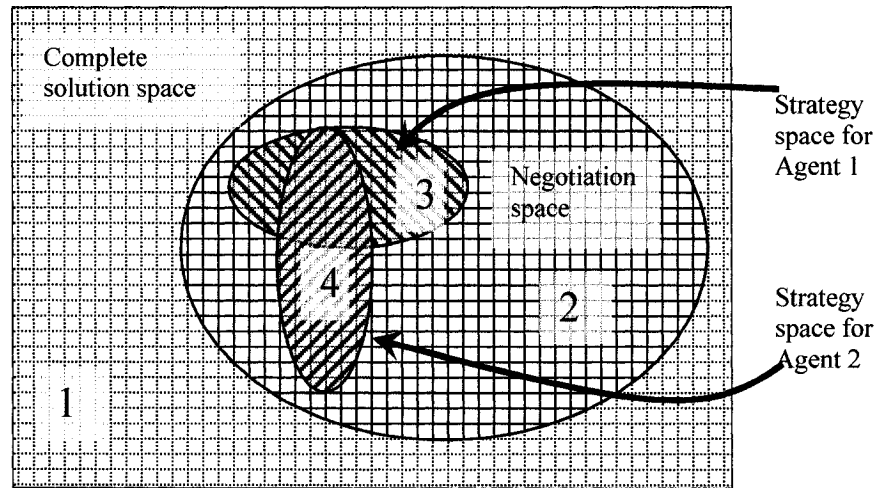
Researchers in MAS have analyzed negotiation protocols that allow for honest and fair allocations. Properties like anonymity, strategy-proofness and stability are emphasized more in MAS. While economists attempt to characterize efficiency in terms of

maximization of social welfare, MAS researchers define it in terms of Pareto optimality. This is because when agents are autonomous and rational no agent will prefer an allocation where it is worse off to make others better off even if social welfare is maximized in the process. Pareto optimality captures this dynamic in agent interactions. This can be visualized by a simple example: Consider 100 apples that need to be divided among 2 agents. Agents 1 and 2 value each apple at \$1 and \$2 respectively. The only possible solution that maximizes social welfare is one where agent 2 gets all the apples while agent 1 gets none ( $sw=\$200$ ). However every solution that allots  $x$  apples ( $x \leq 100$ ) to agent 1 and  $100-x$  apples to agent 2 is Pareto optimal. Thus Pareto optimality tends to maintain some justice (by maintaining status quo) even if the allocation is globally inefficient.

### 3.1. Protocol vs. Procedure

The terms “protocol” and “procedure” are used interchangeably. Generally the term “protocol” is more widely used in the MAS literature on resource allocation, whereas the term “procedure” is more prevalent in classical (mathematical) discussions on fair divisions. The purposes for both terms, however, seem to be quite similar. In general, MAS researchers use the term “protocol” to mean a set of rules, publicly published, which all agents are supposed to follow. “Strategies” on the other hand are plans of actions that can be private and are not assumed to be public knowledge. Thus, based on the “protocol,” the agents can come up with strategies to maximize their own benefit. The objective of mechanism designers is therefore to ensure some minimal properties of the

system are met while giving agents freedom to act for self-benefit. Thus the rules so created influence the feasible set of strategies among which agents can choose. Strategies will therefore always be a subset of the set of actions defined by the rules. See Figure 8



1. a1 and a2 can divide 100¢ in fractional cent amounts.
2. a1 and a2 can divide 100¢ only in whole cent amounts.
3. a1 chooses any whole cent amount greater than 45¢.
4. a2 chooses any whole cent amount greater than 40¢.

**Figure 8.** How strategies are constrained by rules: in this example only whole cent amounts are allowed in the negotiation.

For example, consider the problem where \$1 (100¢) needs to be divided among two agents. Implicitly the protocol may assume that sub-cent values cannot be negotiated (i.e., there is no real equivalent of 0.1¢) and hence only whole cent amounts are allowable in the negotiation. Thus agent strategies can no longer include messages like:  $\langle a1:55.5¢, a2:44.5¢ \rangle$  i.e., agent a1 suggests a division of 100¢ whereby it gets to keep 55.5¢ and gives 44.5¢ to a2. Both agents can only include strategies that propose whole cent amounts (therefore only messages like  $\langle a1:55¢, a2:45¢ \rangle$  are allowable).

In multiagent research a “procedure” can mean a process that may follow up after agent interactions. For example, in Vickery auctions the “protocol” may state that agents submit sealed bids and that the agent with the highest bid will be chosen and will pay the price of the second highest bid. Once agents submit their sealed bids the auctioneer will then run a “procedure” that:

1. Verifies that the bids are valid,
2. Orders the bids,
3. Chooses the highest bid as the winner,
4. Chooses the next highest bid as the winning agent’s payment,
5. Contacts the winner and informs the price due,
6. Enforces payment policy.

However the distinction need not be so clear cut. Mathematicians have used the term “procedure” in lieu of “protocol” to describe their allocation schemes. Thus in case of the divide-and-choose procedure, the rules state that for two agents to divide a cake among themselves satisfactorily one agent cuts while the other chooses. In a multiagent setting these rules can be made part of the protocol (rules that the agents are constrained to follow) as well as the follow-up procedure (a1 is to cut the cake and a2 is to choose a piece). Thus the distinction between “protocol” and “procedure” is arbitrary and a matter of usage. The terms will be used interchangeably in this chapter.

### 3.2. Protocol Engineering

The design of protocols and procedures has been conducted on an *ad hoc* basis. MAS researchers and mathematicians have proposed solutions for very specific problems. In most cases, these solutions though ingenious cannot be generalized to a more general class of problems. Also due to the scattered nature of the literature, analysis of particular protocols/procedures has not been systematic. What is envisioned is that protocol design, development and testing should be conducted in a more thorough manner rather than for solving a specific problem, i.e., a framework for *protocol engineering* should be setup. This will serve two useful purposes:

1. A complete analysis of existing protocols/procedures to enable their extension to more general problem domains
2. A systematic methodology to design new protocols based on the criteria that need to be achieved.

Brams and Taylor [14] have discussed the possibility of constructing allocation procedures that fulfill as many of the interesting criteria as possible. For example, can an allocation procedure be constructed that is envy-free, efficient, equitable and truthful? So far, there seems to be no procedure that is known to fulfill these conditions. Thus there exists a possibility that, mathematically, one may be able to prove no such procedure can exist. Therefore before the process of designing a protocol, designers must first list out the important criteria that the protocol must fulfill and then create one that satisfies the most important ones first. Take the example of designing an auction mechanism for selling perishable goods like fruits. The important criteria that need to be fulfilled are (in

order of priority): time, truthful bidding, envy-freeness, efficiency. Thus creating an auction protocol that terminates quickly (time), is satisfactory to all agents (envy-free), induces agents to be truthful but that is not efficient, may be a reasonable tradeoff to achieve. The protocol design process was outlined by Rosenschein [7] and is formalized in Figure 9. In the following sections I present the various criteria that have been discussed in literature. They are partitioned into five fundamental categories. The categories are:

- Criteria for allocations
- Criteria for protocols
- Criteria for procedures
- Criteria for resources
- Criteria for utility functions

### **3.3. Criteria for Allocations**

For a divisible resource that will be partitioned into portions and allotted to agents, one needs to determine if the agents are satisfied with the portions allotted to them. The following criteria will be used for the purpose:

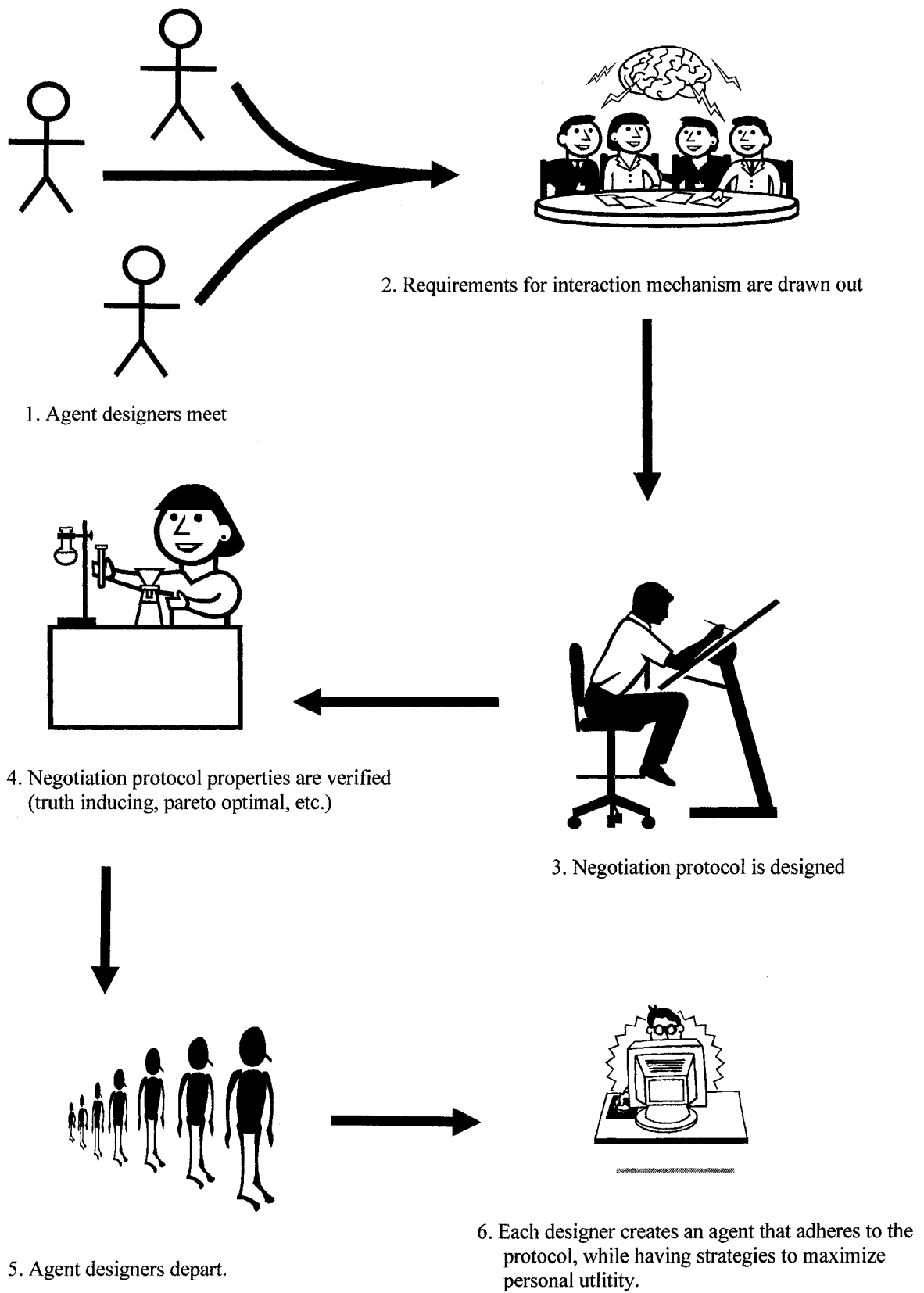
### 3.3.1. Fairness

This perhaps the oldest criteria mentioned in the research of fair division and most commonly used across all disciplines. However the concept behind its usage is diffuse. In some settings fairness is mentioned as a diffuse concept that can be achieved by fulfilling various “fairness” criteria. Brams and Taylor however set it on a more concrete footing. An allocation is deemed to be *fair* if every agent thinks it got at least  $1/n$  of the entire resource allocated. This criterion is fulfilled by almost all protocols (or procedures) that are mentioned in literature. The fairness criterion has been further refined in literature and sub-criteria that qualify fairness have been suggested. The following sub-criteria are mentioned in the increasing order of strictness.

- **Approximate Fair** [31]: This criterion is invoked in cases where getting even simple fair allocations prove to be too difficult. Here each agent is assumed to have a tolerance  $\epsilon$  below which it considers the value of the piece to be negligible. Thus an allocation is *approximate fair* if each agent gets a piece that is **at most  $\epsilon$  smaller than  $1/n$**  of the total resource allotted.
- **Simple Fair**: An allocation is *simple fair* if each agent gets **at least  $1/n$**  of the total resource allotted. This is the commonly accepted definition for fairness.
- **Strong Fair**: An allocation is *strong fair* if each agent gets **more than  $1/n$**  of the total resource allotted.
- **Max-min Fair** [84]: This is not directly related to the fairness criteria described so far. The criterion has been suggested to solve the problem of packet scheduling in communication networks. It consists of distributing a commonly contended resource

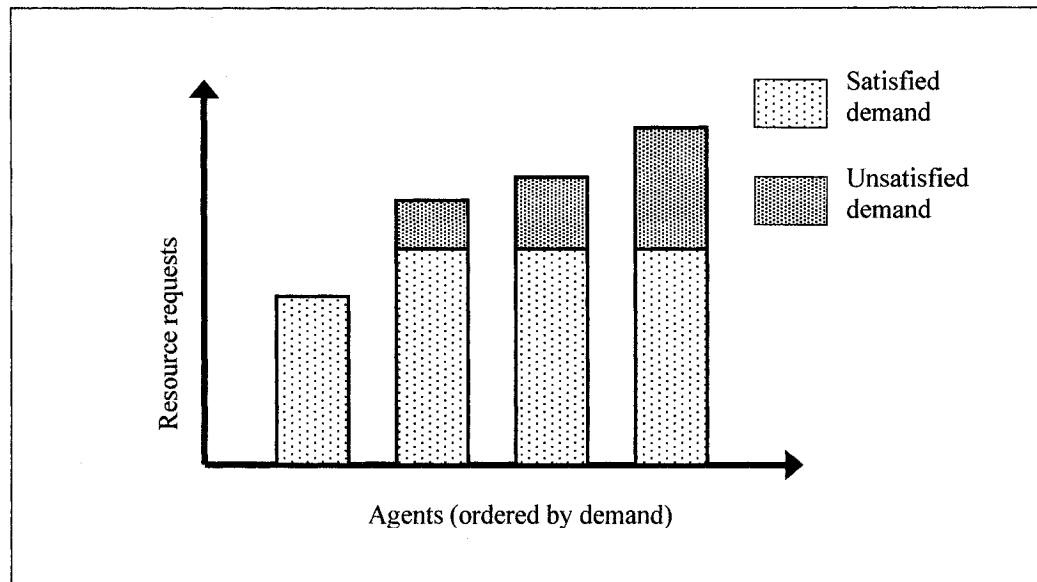
among a number of competing agents. The basic idea is to satisfy as many agents as possible and then distribute any leftovers to the agents which have larger demands.





**Figure 9.** Engineering a protocol.

Thus the demands of the agents are first ordered from lowest to highest. All agents receive an amount equal to the one received by the agent with the smallest demand. The remaining resources are then split equally among the remaining agents. Figure 10 shows the intuition behind the criterion.



**Figure 10.** How max-min fairness works

Thus the purpose of max-min fairness is to maximize the minimum allocated to each agent whose demand has not been fully satisfied. Some salient differences with respect to earlier fairness criteria are that each agent is assumed to get the same utility over the same portion of the resource and the utilities functions are additive in nature.

### 3.3.2. *Envy-freeness*

Envy-freeness is a stricter condition than fairness. Envy-freeness implies fairness but vice-versa is not true. The set of envy-free procedures is much smaller than ones that are simply fair. Depending on the context, fairness may not be sufficient for getting satisfactory solutions. An allocation is *envy-free* if each agent thinks that its piece is **at least as large as** the largest piece in the allocation. Thus all envy-free solutions are fair. The procedures that have been proposed, however, involve some sort of trade-off to achieve this criterion. The most common trade-off is efficiency. Most envy-free procedures involve one agent trimming off pieces created by other agents (to create ties). These trimmings are then inaccessible to all agents till the end of the procedure and are wasted during the allocation process. Other tradeoffs include increased running times (in terms of the number of cuts or comparisons) of the procedure as well as higher complexity of the rules governing agent strategies. Envy-free solutions are still appealing because once an agent considers its piece to be the largest, it will have no further incentive to try to trade its pieces with others or manipulate the process in some other way (by lying for example). Thus fairness may not be a sufficiently strong condition in many real world cases. Similar to the case of fairness, envy-freeness too has been qualified in various ways to get reasonable solutions in different situations. I define the following sub-criteria, roughly categorized in the increasing order of strictness.

### 3.3.2.1. Alternatives to Envy-Freeness

In many domains attaining envy-free allocations may be quite appealing but it may be infeasible to do so due to various constraints on the system. Therefore as a compromise, suggestions have been put forth that weaken the envy-free criteria suitably so that the spirit of envy-freeness is followed as much as possible while making the criterion flexible enough to allow a larger space of feasible solutions.

- **Non-domination** [82]: An allocation is considered to be have “*absence of domination*” if for each agent  $i$  and every other agent  $j$ , there exists at least one agent  $k$  ( $k \neq i, j$ ) that prefers agent  $i$ 's portion over agent  $j$ 's. This means that not everyone in the society of agents thinks agent  $i$  got a smaller portion than agent  $j$ . Thus agent  $i$ 's piece is not “dominated” by any other agent's piece.
- **Egalitarianism** [85]: In some economies it has been shown that among all Pareto-optimal solutions none may be envy-free. Thus there can be conflict in the criteria of envy-freeness and Pareto optimality. Economists therefore put forth egalitarianism an equity criterion that was close to envy-freeness in spirit while being more flexible as well. An allocation is said to be *egalitarian-equivalent* if there exists a piece of the resource (the same for every agent) that is considered by each agent to be indifferent to the bundle it actually gets in the allocation. Thus all envy-free solutions are egalitarian-equivalent but the vice-versa is not true.

### 3.3.2.2. Envy-Minimizing Criteria

In situations where no envy-free allocations can be found researchers have suggested approximations that will do the best that can be achieved. Thus minimizing envy in allocations is a natural extension of the objective of getting envy-free allocations when this objective cannot be achieved. A number of indices for measuring envy of a given allocation have been proposed.

- **Minimal ordered-pair envy** [82]: This index indicates the number of ordered pairs of agents,  $(i,j)$  in which  $i$  envies  $j$ . Thus if  $x$  is the allocation and  $x_i$  is the portion allocated to agent  $i$  which has utility function  $u_i$  then:

$$\varepsilon_x = \{(i,j) \mid u_i(x_i) < u_i(x_j)\}$$

is the set of ordered pairs  $(i,j)$  where  $i$  is envious of  $j$ 's portion in allocation  $x$ . According to the Feldman-Kirman procedure the objective of an allocation should be:

$$\min_x \text{Card } \varepsilon_x$$

i.e., the number of ordered pairs that feel envy should be minimized.

- **Minimal intensity envy** [82]: The minimal ordered-pair envy shown above has no notion of the intensity of envy present. For example, if  $i$  envies  $j$  and  $k$  then does it envy  $j$  as “strongly” as  $k$ ? There is no way to distinguish weak envy from strong envy. The main reason intensity levels are not measured is because it requires interpersonal comparisons of utility which will make the measure meaningless. However there is a way to measure intensity of envy without making such

comparisons. For any allocation  $x$  and any pair of agents  $i$  and  $j$ , let  $\varepsilon_{x,ij}$  be a scalar that satisfies:

$$u_i(x_i) = u_i(\varepsilon_{x,ij}(x_j))$$

Thus  $\varepsilon_{x,ij}$  represents the fraction of  $j$ 's portion which  $i$  thinks is of the same value as its own. After suitable algebraic manipulations it has been suggested that the objective of the allocation should be to minimize  $\xi_x$  where:

$$\xi_x = \sum_{i,j} \lambda_{x,ij}$$

and

$$\lambda_{x,ij} = (1/\varepsilon_{x,ij}) - 1$$

Thus the objective is to minimize the intensity of global envy across all agents.

- **Worst-off Envy** [83]: This is based in Rawl's idea of making the worst-off person as well off as possible. Let  $e_{ij}[x]$  have value 1 when  $i$  envies  $j$  and be zero otherwise.

Then

$$e_i[x] = \sum_{j \neq i} e_{ij}[x]$$

is the number of people that  $i$  envies in allocation  $x$ . The index for worst-off envy can then be defined as:

$$m_{we}[x] = \max_i e_i[x]$$

and the objective is to find the allocation that minimizes  $m_{we}[x]$ .

- **Median Envy** [83]: Median envy measures the number of agents which envy more than half the number of other agents based on portions allocated to them. Based on the terms defined above for worst-off envy, median envy can be defined as:

$$m_{mde}[x] = \left| \{i : e_i[x] > (n-1)/2\} \right|$$

- **Mean Envy [83]:** This index measures the average number of others that an individual envies. It is defined as:

$$m_{me}[x] = \sum_i e_i[x] / n$$

Since it is the average, it ignores the variability of envy present in various allocations. This index may be combined with the Pareto-optimality condition to account for rational preferences agents make. Mean envy is a more refined measure than Median envy and can be use as a secondary criterion to choose among alternative allocations all of which are median envy-free.

- **Approximate Envy-Free [14]:** Similar to approximate fair, the criterion of approximate envy-free is used to “tend to” envy-freeness in cases where getting simple envy-free allocations is not feasible. Let  $\varepsilon$  be a small positive number which is the smallest portion which is of value to any agent. Anything smaller than  $\varepsilon$  is considered to be negligible by the agent. One can call an allocation *approximate envy-free* if no agent thinks its portion is **smaller by  $\varepsilon$  than** the largest portion in the allocation. The notion of approximate envy-free is used to propose many discrete procedures to allocate resources that compromise on the exact envy-free criterion enough to allow them to have efficient bounds on the number of cuts as well the time steps required to complete the procedure. In real world problems, such approximations might be tolerable compared to the alternative of not having envy-free procedures at all. Once  $\varepsilon$  is made small enough, the solution found would be satisfactory for all parties involved.

### 3.3.2.3. Envy-Free Criteria

I begin by defining some terms that will be commonly used in this section. Let  $u_i(x_i)$  denote the utility agent  $i$  derives from the portion allocated to it. Agent  $j$  ( $\neq i$ ) denotes some other agent against whom one compares agent  $i$ 's utility. One has the following versions of envy-freeness in the increasing order of strictness.

- **Simple Envy-Free:** This is the commonly understood notion of envy-freeness. An allocation is *envy-free*, if every agent gets a portion of the resource that is **at least as large as** any other agents' portions.

$$u_i(x_i) \geq u_i(x_j)$$

- **Strong Envy-Free [86]:** This is a stricter condition than simple envy-freeness. An allocation is *strong envy-free*, if every agent gets a portion of the resource that is **larger than** any other agents' portions. Here the agent is not only envy-free; it actually believes that other agents are envious of its portion.

$$u_i(x_i) > u_i(x_j)$$

- **Super Envy-Free [86]:** This is a stricter condition than Strong envy-freeness. An allocation is *super envy-free*, if every agent believes that every other agent has **at most** a fair share of the resource. Naturally here every agent not only believes it got the largest share, it also believes all other agents got a fair share at best.

$$u_i(x_j) \leq 1/n$$

- **Strong Super Envy-Free [86]:** This is a stricter condition than Super envy-freeness. An allocation is *strong super envy-free*, if every agent believes that every other agent has **less than** a fair share of the resource.



$$u_i(x_j) < 1/n$$

- **Opportunity Envy-Free [82]:** This is a suggested measure for fairness criteria that comes from the domain of social justice. It tries to formalize the notion that for equality among individuals to be attained, rather than apportion the resources equally, all agents be given equal means to fulfill their goals. Thus one agent will be envy-free of another if it gets the same means like everyone else to fulfill its goals. The opportunity set  $C_i$  for agent  $i$  is the set of allocations  $x_i$  that are feasible given its initial endowments. Let  $x^*$  be an allocation such that  $x_i^* \in C_i$  for each  $i$ . An allocation is *opportunity envy-free* if for every pair of agents  $i, j$ ,

$$u_i(x_i^*) > u_i(x_j) \quad \forall x_j \in C_j$$

Thus agent  $i$  does not envy the opportunity available to any other agent. Opportunity envy-free implies envy-freeness but the converse is not true. This is a stricter condition, therefore, than envy-freeness.

### 3.3.3. Exact

This can be taken as a special case where many criteria mentioned earlier intersect. An allocation is *exact* if every agent believes **everyone** has received **exactly**  $1/n$  of the total resource allocated. The set of allocations that are exact is the subset of the intersection of the sets of allocations that are approximate fair, fair, egalitarian, approximate envy-free, envy-free, and super envy-free. Dubins and Spanier [43] have mentioned about the

existence of exact allocations given that each agent has a measure function over the entire resource. There are however no constructive methods known to find such allocations.

#### **3.3.4. *Equitability***

This is a criterion put forth by Brams and Taylor [14] for their Adjusted Winner (AW) procedure for allocating a resource. Since AW is a specific two person procedure to divide heterogeneous resources and requires certain assumptions about the properties of the agent utility functions, equitability is also valid only under such conditions. The setup involves two agents each with different utilities for each item in the bundle. Both agents are required to allot points to each item in the bundle, reflecting their relative values to each agent, such that the points sum to 100. Then the equitability condition states that the goods should be divided among the two agents in such way that both agents get goods worth equal number of points; i.e., not only should both agents think they got a larger share than the other; they should also think their share is larger by the same amount. This step requires that agents have utility functions that are additive and linear. Linearity means the agents' marginal utilities are constant. While envy-freeness and equitability both address the question of whether one agent believes it did at least as well as the other agent, the difference is that envy-freeness involves intra-agent comparison of utilities of different portions to a particular agent, whereas equitability involves a more controversial inter-personal comparison of utilities of different agents for a particular portion. Thus the objective is to give each agent portions such that their point totals are the same. However it requires that comparisons between different utility functions be made which may not

hold up in real world scenarios. This criterion can be applied lexicographically after efficiency to obtain better satisfaction for the participating agents.

### 3.3.5. Efficiency

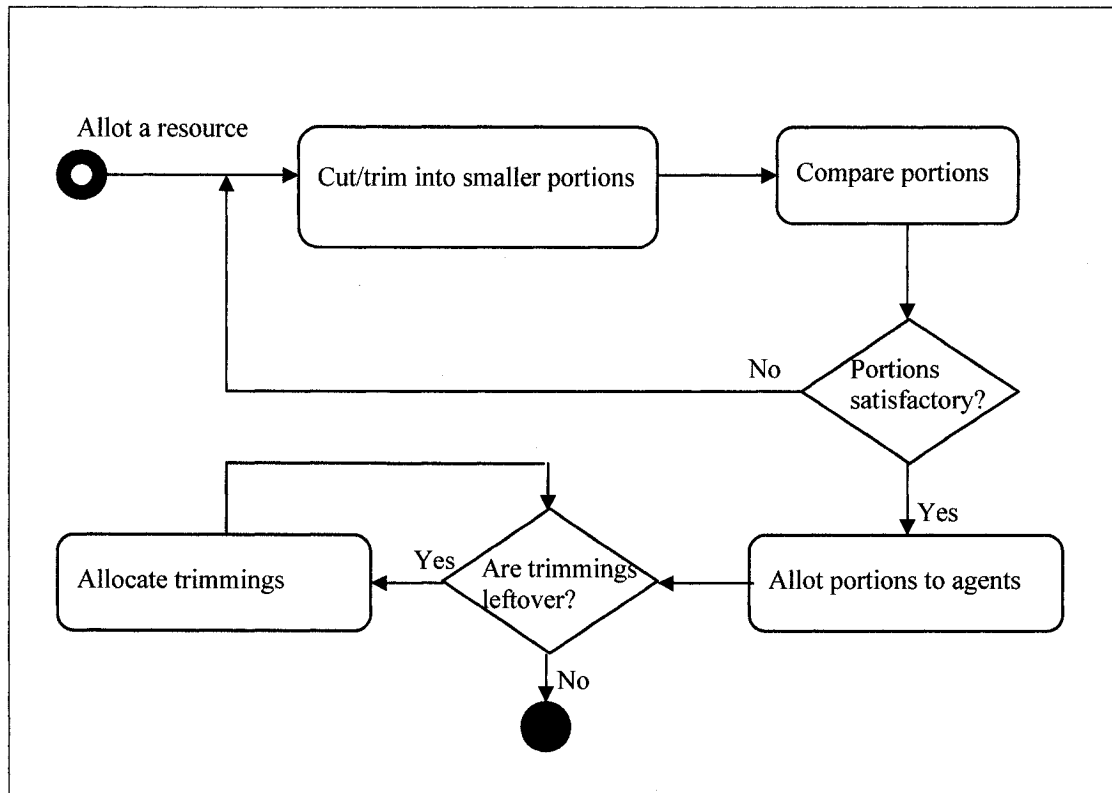
Efficiency is deemed as an important criterion in many allocations right after they are checked for fairness and envy-freeness. Researchers in social welfare and economics sometimes prefer this criterion be fulfilled over envy-freeness. However MAS researchers may find efficiency to be secondary to envy-freeness, strategy-proofness and stability among others. Efficiency criteria roughly indicate whether there has been wastage in the allocation procedure. Wastage mainly happens in envy-free procedures that trim the portions cut out of the resource to create ties for the largest piece. The trimmed portions generally are wasted. Efficiency criteria therefore typically run foul of envy-freeness. Some simple retrofitting of the procedures can reduce wastage. For example, the allocation procedure can be run iteratively, first on the whole resource, then on the leftovers, then on the leftovers of the leftovers and so on *ad infinitum*. Some procedures are however created from the ground up to be envy-free as well as efficient. While this is ideal, the rules tend to be quite complex and almost no procedures exist for allocating to general  $n$  number of agents. There are generally two schools of thought about how to approach efficiency. Those who adhere to a utilitarian theory have generally connected efficiency to the maximization of social welfare, while economists tend to look at efficiency in terms of Pareto optimality.

- Social welfare:** The roots of social welfare are based on utilitarianism theory that seeks quantitative maximization of good consequences for a population. Each agent is assumed to have a utility function that reflects the satisfaction it receives from obtaining a portion of the resource. The utility functions can be distinct for each agent and generally are not comparable to those of another agent. Thus while utility theory can say if an agent is *happier* in receiving one portion over another, it cannot say if one agent would be *happier* than another agent in doing so. Social welfare refers to the overall welfare of society. It is defined as the summation of the utilities of all agents in society. Ironically social welfare tends to let individuals get monopoly over resources rather than benefiting society as a whole. Picking up the simple example mentioned earlier. If 100 apples need to be distributed between agent 1 and 2 who value them at \$1 and \$2 respectively, then social welfare is maximized if all the apples are allotted to agent 2 leaving agent 1 with no share. Thus social welfare functions may be at odds with equity and justice criteria. Since cardinal comparisons between utilities is involved, some economists have preferred to measure efficiency using the Pareto optimality measure
- Pareto Optimality:** This is an alternative proposition for efficiency. It does away with inter-utility comparisons. Rather than utilize cardinal measures of utility, it uses ordinal preferences to rank allocations. An allocation is *Pareto optimal* if there is no other allocation that makes every agent at least as well off and at least one agent strictly better off. This requirement fits well in game theory and multiagent systems where players (or agents) are assumed to be autonomous and rational. It is quite reasonable to assume that given an initial distribution of resources no agent will

agree to any alternative distribution where it has to sacrifice its welfare even if it improves the condition of society as a whole. Pareto optimality thus tends to maintain the status quo. Depending on the perspective however this is seen as desirable or undesirable. Continuing on our previous example of dividing apples among agents, any solution which grants  $x$  apples to agent 1 and  $100-x$  apples to agent 2 ( $0 \leq x \leq 100$ ) is Pareto optimal. The social welfare solution of agent 2 getting all the apples is also a member of the set of Pareto-optimal solutions. Thus social welfare solutions are a subset of Pareto-optimal solutions. As can be seen, Pareto optimality does not necessarily promote equity in distributions and monopolistic allocations are ranked as high as fair ones.

- **Frugality (Efficiency in Number of Cuts)** [14]: A procedure for  $n$  players is called *frugal* if it uses  $n-1$  cuts to complete the resource allocation. All allocation procedures need to make comparisons before coming up with a feasible solution. Since continuous algorithms do not need to cut resources into portions while making such comparisons they are generally very efficient in the number of cuts made. Thus the moving-knife procedure does the job in the least number of cuts ( $n-1$ ) possible. Since at any given time,  $k$  agents (the ones which have not yet received a portion) evaluate the size of the portion to the left of the cake simultaneously and continuously, the number of comparisons made is infinite. Discrete procedures however run in steps and can compare only two pieces at every step. Each comparison is made between portions actually cut out from the resource. Typically the procedures iterate as shown in Figure 11. There is only one discrete procedure known to complete in  $n-1$  cuts. The literature shows that there are many continuous

procedures that are frugal while all but one of the discrete procedures are non-frugal. The divide-and-choose procedure is the only discrete procedure that manages to be frugal because it can accomplish a satisfactory division of a resource among two agents with just one cut. It is worth mentioning that the Selfridge-Conway discrete procedure for 3 agents manages to allocate resources in a maximum of 5 cuts and the 4-agent procedure proposed by Brams, Taylor and Zwicker get envy-free, efficient allocation with a maximum of 11 cuts. However the generalized procedure when extended from 4 to any  $n$  number of agents will allocate only after an arbitrary number of cuts.



**Figure 11.** A general view of how discrete allocation procedures work

- **C-Efficiency (Efficiency in Frugal Procedures) [14]:** The discussion of frugality showed the difficulty of obtaining discrete procedures which can complete allocation

in the fewest cuts ( $n-1$ ) possible. Brams and Taylor discuss the notion of a weaker form of efficiency connected to frugal procedures. An allocation procedure for  $n$  players is called a *C-procedure* if it results in a set of  $n-1$  cuts of a rectangular cake, all of which are parallel to the left and right edges of the cake. One can then define C-efficiency as follows: “A C-procedure for  $n$  players is C-efficient if there is no other C-procedure for  $n$  players that yields an allocation that is strictly better for at least one player and as good for all others,” i.e., a C-procedure that is Pareto optimal is C-efficient. Brams and Taylor use this definition to connect envy-freeness and efficiency. They show that any envy-free C-procedure is C-efficient.

The discussion shows that efficiency is so far seen as a binary variable. An allocation is either efficient or not. This definition is too constraining and is not able to judge procedures which can improve upon efficiency while being inefficient in the traditional sense. There is a clear need for creating a measure of efficiency that can rank procedures based on the amount of resource wasted.

### ***3.3.6. Relationship between Various Allocation Criteria***

The relation between the various allocation criteria is shown in Figure 12. The salient features of the figure are explained below. The set of approximate fair allocations is the largest set that encompasses all the other criteria (except egalitarianism). It can be shown that the set of approximately envy-free allocations is a proper set of approximate fair allocations. However the set of fair allocations is not a proper subset of the set of

approximate envy-free allocations or vice-versa. Consider three agents and let  $\epsilon$  be a small positive number that is the smallest portion that has any value for an agent.

Consider the valuations of the distributions from the perspective of agent 1 (refer Table 1). It is to be noted that the examples show that the portion (not the entire cake) allocated to agent 1 fulfills a particular criterion (say, being approximate envy-free). They are from the point of view of agent 1. In order for the allocation (the entire cake) to fulfill the criteria (of approximate envy-freeness) it is required that all agents believe the said criteria is fulfilled for their respective portions.

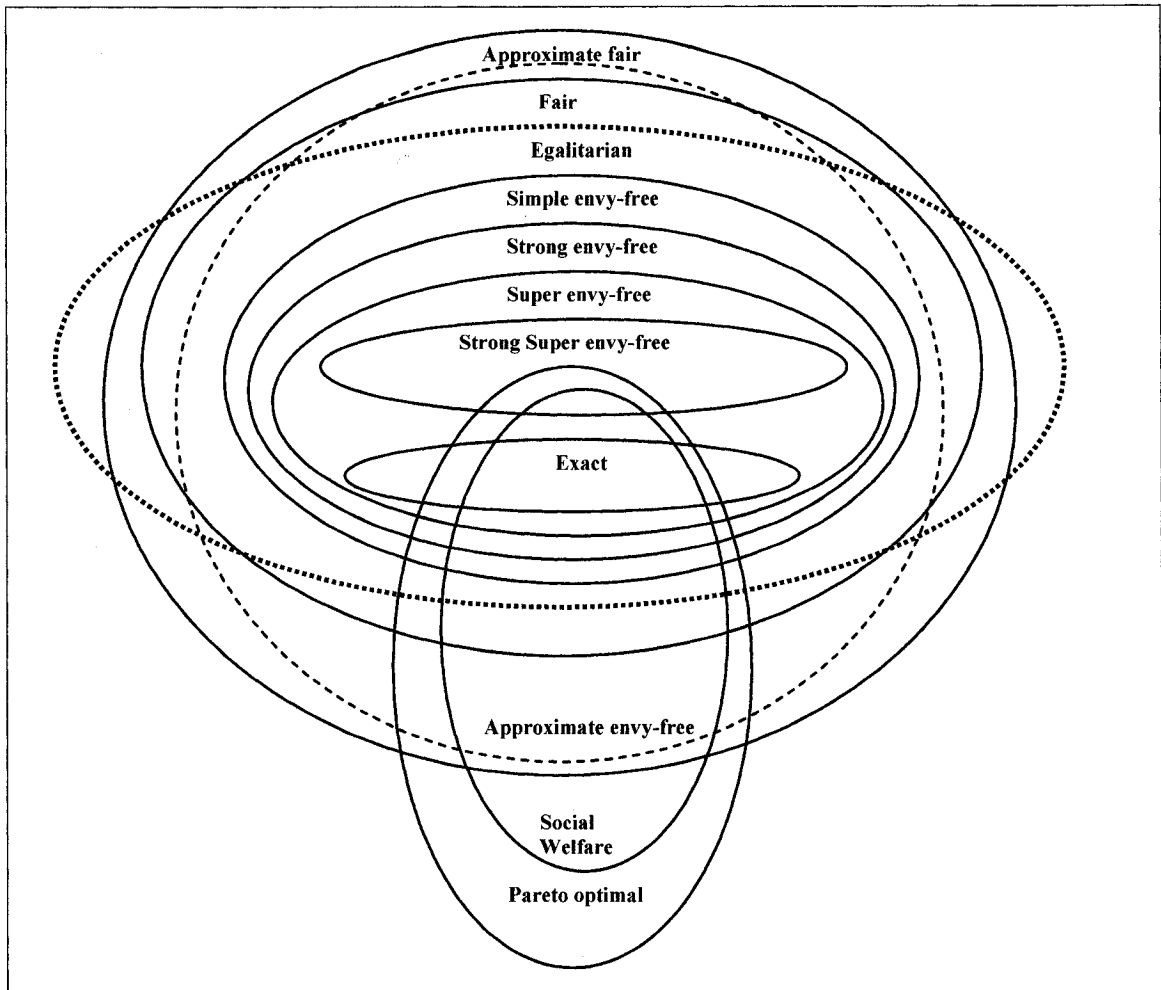
Portion	1	2	3	Approximate fair?	Approximate envy-free?	Fair?
Allocation 1	$1/3 - \epsilon$	$1/3$	$1/3 + \epsilon$	✓	✗	✗
Allocation 2	$1/3 - \epsilon/2$	$1/3$	$1/3 + \epsilon/2$	✓	✓	✗
Allocation 3	$1/3$	$1/3$	$1/3$	✓	✓	✓
Allocation 4	$1/3$	$1/3 - 2*\epsilon$	$1/3 + 2*\epsilon$	✓	✗	✓

**Table 1.** The valuations shown are from the Perspective of Agent 1. The allocations show how approximate fair and approximate envy-free relate to each other.

### 3.4. Criteria for Protocols

This category mainly considers issues of negotiation from the MAS point of view. Rosenschein and Zlotkin [7] formalized the idea of negotiation in MAS. Criteria such as stability, simplicity, and strategy-proofness are of primary concern. I discuss these and other related criteria in this section.





**Figure 12.** Relationships between the various equity criteria.

### 3.4.1. *Symmetric* [7]

An agent creator would not want the negotiation process to be arbitrarily biased against his agent. A protocol is called *symmetric* if it is not biased against any agent for arbitrary or inappropriate reasons. Thus for example, an agent should not be awarded a contract by a contractor because the owner of the agent's owner happened to be a close relative of the contractor. Exactly what constitutes as inappropriate criteria depends on the specific

domain. A good way to ensure symmetry is to hide the identity of the agent from the other parties involved in the transaction.

### **3.4.2. *Simplicity* [7]**

Protocol designers should endeavor to create simple protocols that make low computational demands on agents and minimize the communication overhead. If protocols encourage very unpredictable behavior among agents, then agent designers will be encouraged to create agents that model their opponents and attempt to compensate for such behavior. This may however induce more unpredictable behavior because the opponents' models will be imprecise. Thus not only will each agent be burdened with more computation, overall system efficiency may be lowered as agents waste resources trying to double guess each other. Simplicity, therefore, becomes a strong normative requirement. Simplicity can be derived if protocols are stable. Since the optimal outcome is well known, there is nothing better to do than just carry it out.

### **3.4.3. *Stability* [7]**

Systems that are stable will reduce agent computational load and communication costs. From the game-theoretic point of view, stability means systems that have at least one Nash equilibrium. A system is *stable* if no agent can do better by moving away from its current strategy while other agents maintain their current strategies. Thus protocols need

to be created that encourage agents to keep strategies that are in equilibrium with each other. Stability cannot be enforced but can only “emerge” voluntarily from agents, because it is assumed that agents are autonomous and rational. Since the agents are non-cooperative the protocols need to induce stable behavior from them so that the entire system can behave predictably.

#### ***3.4.4. Anonymity and Privacy***

Anonymity deals with the issue of keeping agent identities hidden for negotiation in open systems. The Internet is an open network and agents may desire privacy through anonymity. It also protects agents from being profiled by opponents who can then use the information for future negotiations. Open-cry auctions tend to reveal the identity of agents since they are held publicly. In an English auction, for example, every agent’s true utility is revealed except the winner’s. This is because every rational agent will stop bidding once price of an item has exceeded the true utility for that agent. Anonymity can be assured in a sealed bid setting however. The auctioneer need only inform every agent (that is not the winner) that it has lost. The winning bidder and bid need not be communicated to everybody. Thus anonymity improves privacy of personal agent information. Privacy deals with the issue of keeping private agent information hidden from competing agents. Knowledge of an opponent’s utility can be put to advantageous use. Privacy is therefore an important criterion in negotiation protocol design.

### **3.4.5. Strategy-Proof [7]**

This is another game-theoretic criterion. A protocol is said to be *strategy-proof*, if for each agent declaring its true evaluation is a dominant strategy. A typical example of a strategy-proof mechanism is the Vickery auction where the protocol requires that agents submit sealed bids to the auctioneer and the agent quoting the highest price wins but is only required to pay the amount of the second highest bid. Alternatively, the term incentive compatibility has been used. A negotiation mechanism is called *incentive compatible* when the strategy of telling the truth (or behaving according to one's true goals) is in equilibrium. Strategy-proof protocols lead to outcomes where lying by a particular agent will only negatively affect its own utility while all other agents do no worse than getting what they would have received if the agent were truthfully revealing its utility. While strategy-proof mechanisms allow fairer and more efficient outcomes, it may not be preferred by participating agents because it tends to make their private information public, thus making them more prone to manipulation in future negotiations.

### **3.4.6. Centralized vs. Distributed**

As it happens, while multiagent systems are discussed as an example of distributed systems, the interaction protocols that are designed for sharing resources are centralized in nature. Distributed protocols are generally more difficult to create and many protocols suggested may involve some degree of centralization at subtler levels. The main drawback of centralized protocols is that the system is prone to a single point of failure.

Thus in any auction mechanism, if the central auctioneer goes offline, the entire system goes down. Distributed negotiation schemes are by contrast less susceptible to failures due to any particular agent going offline. They are fault tolerant. In order to achieve a decentralized protocol however, some redundancy needs to be built into the system in order that other agents may assume the role of the failed agent. Thus redundancy is required in terms of roles, resources and information. The trade-off is that distributed systems may arrive at less globally efficient outcomes and take more time to reach stability. However distributed systems can also protect the privacy of agent information better. The distributed model seems more natural in cases where finding optimal allocation may be computationally infeasible. Even small improvements over initial allocation of resources would be considered a success. Stepwise improvements over the status quo are naturally modeled in a distributed negotiation framework. The English auction or the Dutch auction are examples of centralized systems whereas the contract net protocol is an example of a distributed mechanism.

#### ***3.4.7. Mediated vs. Arbitrated***

Arbitration is a process in which contending parties submit their dispute to an arbitrator who is an impartial person rendering judgment on how the dispute is to be resolved. The judgment is then binding on the parties involved. The arbitrator is expected to be of high repute and unbiased. Arbitration is the process involved when disputes are settled through the legal court system. In multiagent societies an auctioneer can be considered an arbitrator. While most auctions have clear criteria on winner determination (lowest bidder

wins in a call for proposal), if various other properties need to be considered like swift job completion, quality of service etc. the winner will depend on the opinion of arbitrator. This is used less often in multiagent systems due to the inherent subjectivity involved in judgments. Mediation, on the other hand, involves an unbiased person who *facilitates* the settlements of disputes. A mediator does not pass judgment or opinions. Rather he clarifies the rules of the negotiation and acts as a catalyst between opposing interests. The person holding the knife in the moving-knife procedure is performing the role of a mediator. Arbitration is adjudicatory as opposed to mediation which is advisory. Procedures that require either a mediator or arbitrator suffer from the weakness that the bias (or lack thereof) of the arbitrator / mediator cannot be proven definitively. Generally bilateral negotiation protocols do not need any mediators. Multilateral negotiation, on the other hand, feature some form of a mediator, who sifts through the various offers, to come up with an allocation satisfactory to all agents.

#### **3.4.8. *Communication Complexity [87]***

Communication complexity is more of a concern for MAS researchers than mathematicians or economists. Centralized protocols tend to be more efficient in communications (since all agents have to talk only to the auctioneer) than distributed ones. Similarly, one-shot protocols communicate more frugally than iterative protocols. Endriss and Maudet have defined communication complexity of a protocol as “the maximal number of bits exchanged when following the protocol (in the worst case).” If the costs of arranging a multilateral deal is proportional to the number of pairs in a group

of agents then communication costs are of the order  $O(n^2)$ . If the costs are proportional to the number of subgroups in a group, then the communication costs would rise exponentially  $O(2^n)$ . The communication costs are studied as the combination of the number of deals and communications to agree on a deal that could be needed to finalize an allocation. Endriss and Maudet have given a detailed analysis of the communication complexity for various classes of utility functions that allow any rational deal to be implemented.

#### **3.4.9. Time [26]**

Time has not been studied seriously as a criterion for negotiation protocols. This is because the environment is assumed to be static when the negotiation takes places. The assumption can be wrong in many ways. If the resource is perishable, then agents can benefit more from a quick but inefficient protocol rather than a lengthy but efficient one. If by the end of the negotiation, the resource has lost its utility by a significant amount then the objective of profiting from ownership of the resource is defeated. There are other time constraints possible. If agents count the time required for negotiation as time lost in following up other potential opportunities then it makes sense to keep the negotiation short. Also, the resources may be required for accomplishing a specific task that has a completion deadline. A negotiation protocol is useless if it achieves a good bargain in acquiring a good but the agent no longer needs it. Agents' attitudes towards negotiation time directly influence the kinds of agreements they will reach. Kraus, Wilkenfeld, and

Zlotkin [88] have studied this issue and shown that the agreements can be reached without delay. Thus the inefficiency of negotiating in many steps is avoidable.

#### **3.4.10. One-Shot vs. Iterative**

A majority of the protocols put forth in the literature are iterative. Iterative protocols allow the gradual discovery of other agents' valuations. Thus each agent can adjust its offer according to the latest information it has on the other agents' bids. This allows for efficient outcomes. The open-cry English auction and Dutch auctions are examples of iterative protocols. Such protocols can however be costly in terms of communication overhead. They also require more time to complete. The one-shot negotiation protocols offer an advantage in this respect. Examples include the first price sealed bid and the second price sealed bid (also called Vickery auctions). Only one round of messages are sent by the agents, after which a mediator or auctioneer decides who wins the resource. One-shot protocols can lead to inefficient outcomes because agents do not have any more chances to adjust their offers after knowing their opponents bids. They, however, protect the privacy of agent bids. Rosenschein and Zlotkin [7] has proposed the one-step protocol as an alternative to multi-step Monotonic Concession Protocol. Since it is guaranteed that the Monotonic Concession Protocol will lead to a stable outcome, Rosenschein proposes that the agents jump all steps and mutually agree on the final outcome in the first step itself. In such cases, a one-shot protocol is as efficient as the iterative protocol while having very low communication and computational needs. Thus, if the domain is uncertain and dynamic then iterative protocols need to be used to allow agents to



“discover” alternative allocations. On the other hand, domains that are simple and well known allow construction of one-shot protocols.

### **3.5. Criteria for Procedures**

Mathematicians have generally dominated in providing procedures or “recipes” for allocating resources. Their approach to solving fair division problems bears remarkable similarity to the way MAS research characterizes agent preferences. Many assumptions explicitly made in MAS (such as rationality and autonomy) are implicitly considered by mathematicians while suggesting resource allocation procedures. Other criteria, such as envy-freeness and efficiency that generally belong in the domain of economics, are considered explicitly. Economists suggest the criteria in a normative tone or to show existence results, whereas mathematicians generally try to “bake” the criteria into their solutions. In both cases, however, the semantics of the terms are the same. Criteria such as envy-freeness and efficiency are therefore not revisited, as they have been discussed earlier. The criteria that follow are more exclusively focused on evaluating procedural aspects of resource allocation.

#### ***3.5.1. Bounded vs. Unbounded Number of Agents***

Is the procedure for resource allocation applicable to  $n$  agents, in general? This is an important criterion for evaluating applicability of procedures in real world situations. The

well known divide-and-choose methodology [14] is applicable to only two agents. While it has many nice features like simplicity, envy-freeness, Pareto optimality etc., its biggest limitation is that it applies to exactly two-agent systems. Procedures [89] have been suggested that extend to an unbounded number of agents while trying to preserve the essence of this solution but invariably they end up compromising on at least one of the features. Most commonly they tend to lose envy-freeness (while still being fair) and/or efficiency. They also tend to be more complex. Thus to get procedures to scale to general  $n$  number of agents a number of compromises have to be made. The literature survey shows specific envy-free, efficient solutions for  $n=3$  by Stromquist [35] and Levmore-cook [14]) and  $n=4$  by Fink [69]). However the only known envy-free, efficient procedure for any  $n$  is by Brams, Taylor, and Zwicker [37]. It is an ingenious solution based on the notion of “irrevocable advantage.” It however is quite complex and involves an arbitrarily large number of cuts. It is to be noted that there exist many alternative procedures if only fairness is demanded. The simplest example of an  $n$  agent fair procedure is the moving-knife procedure which is also Pareto optimal. Banach and Knaster [63] have suggested a discrete procedure, based on trimmings, that is fair. However it is inefficient.

### **3.5.2. Discrete vs. Continuous**

This is another important distinction that needs to be considered. In general discrete procedures are algorithmic and therefore implementable on a computer (they may however need plenty of inputs from every agent to discern their preferences). On the

other hand it is not clear how continuous procedures can be used in an automated fashion. A number of issues come into play if a machine wields the knife (say, in the moving-knife procedure). For example the continuous resource may need to be discretized. The moving knife's position may be stated as the number of millimeters from the leftmost point of the cake. However there can be ordering issues if two or more agents call cut at the same millimeter length. This case never arises in original procedure because the resource was continuous and infinitely divisible (hence calls from two agents can be strictly ordered). Another issue is that of synchronization of agent messages. The moving-knife procedure is basically analogous to the English auction. So it is essential that the time order of agent messages be preserved. Unrelated externalities like network congestion can affect the ordering of the bids. Thus an agent with a slow computer or internet connection might find its message reaching the mediator later than another's, although it bid earlier. Many of these issues can be obviated if discrete procedures are implemented as they are not dependent on the time order of messages but carry out the processing in "steps." They are also more amenable to analysis. Thus various features like time and space complexity can be studied to find the discrete procedure most suited to the problem at hand. The major difference is that evaluation of portion sizes by all agents takes place simultaneously in continuous procedures but are spread out as binary comparisons in discrete procedures. Consequently the running time for discrete procedures is much longer than for continuous ones. The number of cuts required is similarly larger in discrete procedures. This is because for evaluating each portion, continuous procedures "demarcate" the portion by the position of the knife. This is not

true in discrete procedures. Thus many continuous procedures have nice features like frugality and C-efficiency.

### **3.5.3. External Payments**

Most procedures that have been proposed involve agents redistributing resources among themselves. There are no external monetary compensations involved. This is quite true in cases where there is a single resource that is infinitely divisible. In such cases the procedure can make exceedingly thin slices of the resource to create equitable divisions. However in some practical situations, the problem may consist of a bundle of heterogeneous resources that are indivisible. A typical example is the problem of divvying up inheritances like a house, car and gold among heirs. Since the house and car are indivisible resources any procedures that apply to divisible resources cannot apply. But if one assumes an extra pool of cash available to each of the agents then it is possible to arrive at envy-free allocations. Knaster's proposal [63] suggests an internal bidding procedure where each heir places bids for each item reflecting their relative value in his or her eyes. The agent that bids highest wins the item. To achieve envy-freeness, the payments made to a common pool are divided among the agents so that the total of the object value and money value is the same for every agent. Procedures incorporating external payments can allow for envy-free and efficient division of resources. They do so by making money the "divisible" resource to work around the indivisibility issue. The obvious drawback of such schemes is that agents are required to have an extra pool of cash available to make payments. Brams and Taylor put forth two point allocation

schemes that do away with cash when dividing indivisible resources. The implicit assumption is that there exists at least one divisible resource along which envy-free and efficient allocations can be created. An additional difficulty is that they cannot predict ahead of time (before agents reveal their true utilities) as to which resource is the one that has to be divisible.

#### 3.5.4. *Verifiability* [49]

Iyer and Huhns [49] have proposed the notion of verifiability to evaluate procedures. This criterion is used to determine if an allocation generated by a procedure is reproducible or not. Due to their essential nature, none of continuous procedures are verifiable, i.e., if a different agent runs the procedure again it will not be able to get the same division of resources. Formally, a procedure is said to be *verifiable* if the allocation of the resource is invariant to the bias of the mediator or the agent. Thus any agent that runs the procedure must get the same results as the mediator. In such a case mediator bias does not play any role. This is quite unlike the case of continuous procedures like the moving-knife where the procedure is vulnerable to manipulation by the mediator. Therefore, no special qualifications are required of the mediator if the procedure is vulnerable. Such procedures are therefore self-mediating and do not require an outsider (with reputation and unbiased judgment) to divide the resource. If procedures are verifiable, disputes can be quickly settled by simply executing the procedure once again. Unfortunately, auctions, sealed bids and most discrete procedures are not verifiable. Iyer and Huhns have proposed an  $n$  agent allocation procedure that is fair, unbiased and verifiable.

### 3.5.5. Computational Complexity

A lot of effort has been expended by mathematicians in coming up with procedures that fulfill criteria like envy-freeness and efficiency. But little has been mentioned about the running time of various procedures. Continuous procedures are not generally studied because there is no clear way to implement them on a computer. As a basic observation it can be said that most continuous procedures involve some form of a moving knife which moves at a constant rate from one end of the cake to the other. A cut is made whenever an agent calls cut. I do not consider the operation of cutting to be a significant parameter in time calculations (although the number of cuts is useful metric). Thus in continuous algorithms only the time taken by the knife to traverse the entire cake is to be considered. Thus the time for execution of the algorithm is independent of the number of agents involved and can be taken to be a constant. I emphasize more on discrete algorithms which are programmable in a computer and are amenable to traditional complexity analysis. There have been attempts [38] to determine the minimum number of cuts required to divide the resource fairly. A given (discrete) resource allocation procedure can be evaluated on two aspects: space complexity and time complexity.

Space complexity deals with the number of comparisons that a procedure needs to make in order to come up with a satisfactory allocation. Space can be thought of as the amount of memory needed to store information about various portions of the resource in order for comparisons to be made. One can therefore define space as the number of portions whose information needs to be held in order for the allocation to be determined. Magdon-Ismail,

Busch, & Krishnamoorthy, show that sorting can be reduced to cake cutting, i.e., any algorithm that performs fair division can sort. Since it is known that sorting algorithms generally take  $\Omega(n \log n)$  comparisons, they deduce the same lower bound on computational complexity for resource allocation procedures as well. For envy-free procedures they determine the complexity to be  $\Omega(n^2)$ . While *all known* procedures are verified to have such properties, they are unable to confirm if *every* procedure will have such properties.

Time complexity roughly corresponds to the number of “steps” an algorithm takes to compute a feasible allocation. This is generally dependent on the number of binary comparisons required which is in turn dependent on the number of cuts. Since all the procedures studied so far are centralized in nature the time complexity corresponds to the space complexity for all (discrete) procedures.

### **3.5.6. Division Independence [50]**

Chambers [50] has put forth the property of division independence as a normative requirement of any division procedure. An allocation procedure satisfies *division independence* if the union of the portions allocated to an agent from various sub-parcels of a parcel is identical to the portion allocated from the initial parcel. The basic appeal for this property is if the total amount of resource available is initially unknown. Thus the allocation procedure can be run on the fly with each new increment to the resource with the guarantee that the combined portions so created will be the same as if the complete

resource were initially available. Another appeal of such a property is that one can potentially create a distributed version of the procedure that be run in parallel on small chunks of the resource since the resulting combined parcels would be the same allocation as the centralized version would have found. Unfortunately Chambers has shown that this criterion negates a lot of other necessary criteria like fairness and allows only dictatorial rules to exist. Division independence is therefore too constraining to be applied to realistic situations.

### **3.6. Criteria for Resources**

Mathematicians refer to resource allocation as the “cake-cutting” problem. The resources are assumed to be like cakes and the problem is to divide the cake (resource) satisfactorily among  $n$  agents. This is because cakes are assumed to be divisible and recombinable. If a good like a house or a car needs to be distributed among heirs, then most procedures that were designed for “cake-cutting” are not applicable. That is because such goods are indivisible. I describe below the various criteria that can be used to discover the kind of resource one is dealing with and hence appropriately devise allocation procedures for them.



### 3.6.1. Divisibility

Divisibility pertains to whether the resource is continuous or discrete in nature. Examples of indivisible resources include a house, car or jewelry that may need to be divided among heirs equitably. The constraints for procedures that allocate indivisible resources are stronger. A resource is called *indivisible* if it can be handed out only in integer units. Fractions of such resources will have no value for the agents. Divisibility on the other hand means that the resource is continuous and portions however small can be created. This implies that the resource is *infinitely divisible*, i.e., any portion, however small, holds a finite value for every agent. In general, it is easier to find solutions for divisible resources rather than indivisible ones. For indivisible resources, a common trick that is employed is to append a divisible resource (like money) and to allocate them together, such that each agent believes it got a fair share. Any procedure that can allocate indivisible resources can work for divisible resources as well. All that needs to be done is to create parcels of a continuous resource that can be treated as discrete units of a (divisible) resource. Thus procedures for indivisible resources are a proper subset of the ones for divisible resources. A further qualification of divisible resources is whether they are recombining or not. If a resource is *recombinable* then a portion that is constituted from many smaller parts will hold a finite value to every agent. That is why cakes are the most commonly used paradigms for discussing resource allocation procedures. Cakes are assumed to be both infinitely divisible and recombining. An example of a resource that is infinitely divisible but not recombining would be the time available on a common CPU for scheduling jobs. While ever finer time slices can be created, two non-contiguous

time slices cannot be recombined. Therefore it is preferable for the agent to receive a large contiguous chunk of time rather than many small scattered chunks of time on the CPU due to the overhead of starting and stopping jobs. Table 2 shows examples of various categories of resources.

Problem domain properties	Infinitely divisible	Combinable
A single row of seats in a theatre	--	--
Time scheduling	X	--
Circular cake	X	X

**Table 2.** Various Problem Domain Types

### 3.6.2. *Homogeneous vs. Heterogeneous*

Resources are *homogeneous* if they are composed of the same kind of material in every part. A cake covered entirely with chocolate icing is considered homogeneous. Barring external issues (like is it the edge of the cake or the center?) every agent is then supposed to value every part of the resource as having the same value. *Heterogeneous* resources on the other hand will have different proportions of materials in different parts. A cake, one half of which is covered in chocolate icing and the other half in vanilla is considered heterogeneous. Agents value the different parts of the cake differently. Most procedures assume the resource to be heterogeneous.

### 3.6.3. *Sharable or Not* [52]

A *sharable* resource can be allocated to a number of agents at the same time. An example of a sharable resource would be a picture taken by a satellite that can be allocated to several different agents at the same time. In general any resource that is some type of information that needs to be transmitted can be shared simultaneously between several agents. Then the only constraint is on the equipment (for example, the camera that generates the picture) that needs to generate this information. Typically since most resources (unlike information) are exhausted on being shared, the canonical sense is that resources are non-sharable.

### 3.6.4. *Measurability* [47]

Measurability is a notion with basis in set theory. A *measure* is a function that assigns a number representing a "size" or "area" or "volume" to subsets of a given set. Such a function therefore has normative requirements to fulfill:

1. It should be non-atomic, i.e., the empty set should measure zero.
2. The measure should be countably additive.
3. Monotonic: The measure function should be non-decreasing in value.

A simple example is the counting measure defined as  $\mu(S) = \text{number of elements in } S$ . Measurability is more of a theoretical concern which does not affect the domain of resource allocation. It can be safely assumed that all real world resources are measurable.

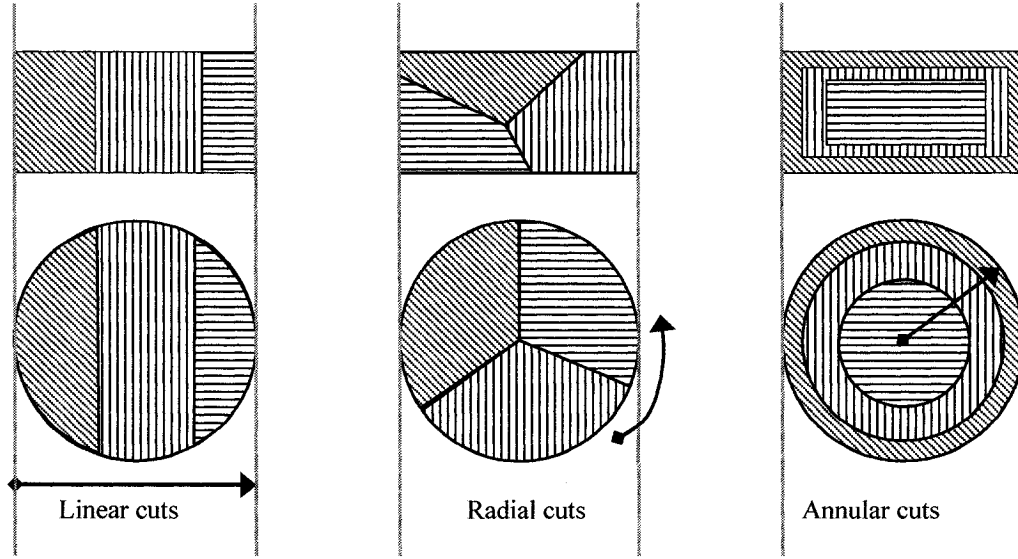
### 3.6.5. Dimensionality

An implicit assumption in the literature of fair division is that all resources are one-dimensional. The dimensionality of resources has never been considered as a parameter. Although cakes are three-dimensional, all procedures assume the knife, that cuts the portions, is parallel to one of the axes and that all agents agree on the same axis. There is a possibility therefore that more efficient allocations exist when the knife moves along one axis than another. But they may not be found because the cake is essentially looked upon as a one-dimensional resource. The literature on land division however shows that resources like land have been considered as two-dimensional resources. However the procedures begin allocation by transforming the two dimensions to one. For example, consider a piece of land contended by many agents. If an agent marks a rectangular portion as its region of interest, the obvious treatment would be to calculate area of the rectangle and compare it to areas of other agents' rectangles in order to come up with a suitable allocation. Since "area" might not be a common measure for all agents, some procedures allocate resources based on how much utility an agent gets from a portion of the (two-dimensional) resource. The process of transforming a two-dimensional preference into one dimension is therefore passed on to the agent, i.e., the agent evaluates its preference for a two-dimensional portion by labeling it with a one-dimensional "utility." Thus even where two-dimensional resources have been considered, the essential allocation is one-dimensional. A procedure for allocating two-dimensional resources has been proposed by Iyer and Huhns that calculates the allocation based on both dimensions. There is no discussion found in the literature for allocating resources of 3 or higher

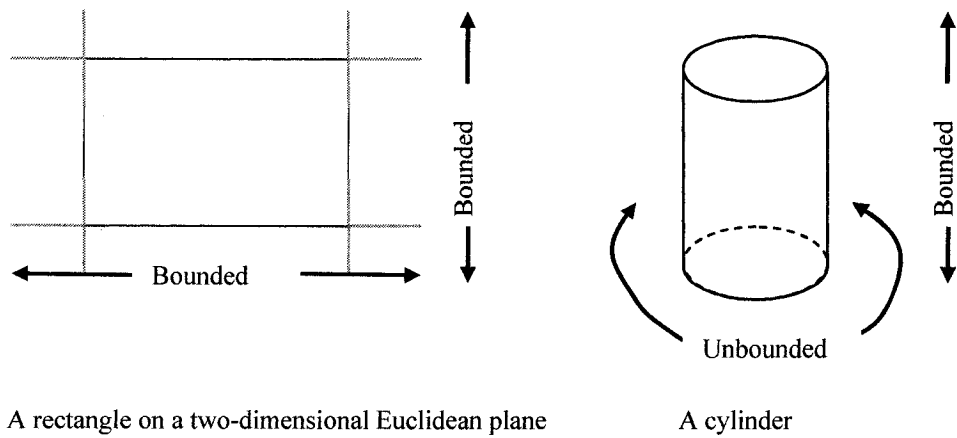
dimensions. It should be noted that dimensionality of a resource need not refer just to the physical dimensions of the resource. *Dimensionality* of a resource is the number of independent variables along which a resource can be defined. Thus an example of a two-dimensional resource allocation problem is the allocation of bandwidth from a common internet connection to various agents. One dimension can be the bandwidth of the connection while the other dimension can be the hours of day when it is needed. Obviously allotting resources with their natural dimensionality in mind can increase efficiency of allocation.

### **3.6.6. Topology**

*Topology* is the qualitative study of geometric shapes. The geometric shape of an object is not the primary focus. Rather topologists study the features of objects that are invariant under continuous deformation. Thus, topologically, a sphere, a cube and a tetrahedron are the same (no holes in the solid, hence homeomorphic to a 2-sphere). As it turns out the topology of the resources is an important factor that affect the quality of an allocation. This is an aspect that has been overlooked in traditional literature. By default, the implicit assumption is that all resources are like an infinite flat two-dimensional Euclidean plane. This is true of the typical cake cutting domain. Thus the portions allocated are linear slices of the cake. The circular cake has been studied too. The circular cake exposes procedures that are topologically different. Thus many circular cake cutting procedures cut radial slices rather than cutting linearly. Figure 13 shows how cake cutting procedures can approach the apportioning of resources.



**Figure 13.** Cake cutting using different topologies for rectangular and circular shaped cakes. The arrows show the direction of knife movement. The figure shows example allocations for three agents.



**Figure 14.** Different resources can have different topologies. Procedures should be designed with these topologies in mind to generate more efficient solutions

Unlike a linear cake, the circle has no endpoints making it topologically different. Hence procedures arbitrarily mark one point on the circumference as the start and end point. Then a convention has to be chosen for the direction of rotation. Similarly, in two dimensions a flat rectangular paper has a different topology than a strip of paper wrapped around a cylinder (as shown in Figure 14). This is because the cylindrical surface has one dimension unbounded and the other bounded. One has options to create procedures that take advantage of the different topologies. There is therefore a strong appeal to study the topology of the resources before creating allocation procedures.

### 3.6.7. *Resources vs. Tasks (Desirable vs. Undesirable) (Goods vs. Bads) [52]*

Tasks, or chores, can be considered to be resources that give negative utility to agents. Resources typically are of positive utility to agents. Thus agents will try to maximize the resources they get while minimizing the tasks they have to perform. Criteria like envy-freeness can be suitably modified to account for tasks. Thus, an allocation is *envy-free* if each agent thinks that's its piece is **at least as small as** the smallest piece in the allocation. Research into specific task allocation procedures is scant because it is generally assumed that tasks are dual to resources. However, an important characteristic of tasks is that tasks are often coupled with constraints regarding their coherent combination. To begin a task, the completion of another task may be required as a precondition. Thus, clearly, procedures for resources allocation cannot be simply retrofitted for task allocation, for all kinds of tasks.

### **3.6.8. *Single vs. Multiunit [52]***

Multiunit resources are not individually distinguishable and agents should not be able to discriminate one unit from another. They have exactly the same features and characteristics. In such cases, allocations and agent preferences can be represented in a compact manner. However more expressive languages are required to represent different quantities of the resource. Single unit resources are individually identifiable and agents can discriminate between units due to their different characteristics. Any multiunit problem can be transformed into a single unit problem by applying unique labels on each of them. The vice-versa is also true; A single unit problem can be converted to a degenerate multiunit problem. However information is lost in the process and subsequent multiunit allocation procedures applied to the resource can yield inefficient allocations.

### **3.6.9. *Static vs. Perishable [52]***

Perishable resources lose their value over time. Food is a good example of a perishable resource. Some protocols like the open-cry descending bid auction (Dutch auction) evolved to quickly trade in perishable items like flowers. Normally protocols designers assume the resource to be static, i.e., the resource does not degrade or diminish with time.



### **3.6.10. Consumable vs. Resusable [52]**

Resources like fuel, foods etc. are consumable. Once the resource has been made use of, it cannot be used again. Reusable resources are available to use after being put to work for a particular agent. The distinction as to whether the resource is consumable or perishable can be a matter of perspective. A satellite (used for weather imagery) is reusable but the specific time slot when it is needed (say 15<sup>th</sup> February 2005, from 9am to 10 am) is not.

### **3.7. Criteria for Utility Functions**

The literature survey shows that a number of assumptions have been made, on the type of utility functions that agents can have, in order for allocation procedures to work. It is unknown if the properties of the procedures hold when agents' utility functions do not respect these assumptions. I mention the typical restrictions on agent utility functions mentioned in literature:

#### **3.7.1. Non-Atomic**

This requirement specifies that the agent utility functions smoothly go down to zero as the amount of the resource tends to zero, i.e., there should be no "atoms" in the resource, where portions smaller than the "atom" are of zero value to the agent.

### 3.7.2. Additive

This is common requirement for utility functions and helps simplify a lot of analysis. If A and B are two disjoint subsets, then simple additivity states that:

$$v(A \cup B) = v(A) + v(B) \text{ for all disjoint } A, B \in \Sigma$$

### 3.7.3. Concave

Concave domains are a subset of subadditive domains and the utility functions have the property:

$$v(A \cup B) \leq v(A) + v(B) \text{ for all disjoint } A, B \in \Sigma$$

They are less commonly modeled than simple additivity because it is difficult to prove results in this domain. However they more realistically reflect the utility of obtaining additional resources in the real world.

### 3.7.4. Continuous

Utility functions are assumed to be continuous. This means that for every amount  $x$  of the resource the agent possesses a unique value:  $v(x)$  is assigned for that amount. This simplifies analysis but may not be necessarily true in the real world. In addition, utility

functions are said to be smooth if there are no “kinks” in the function value anywhere, i.e., the function is differentiable at all points of the curve.

# Chapter 4

## Protocols and Procedures for One-Dimensional Resource

### Allocation

A novel solution, for  $n$  agent, one-dimensional linear resource allocation problem is proposed. This protocol enables agents to choose portions based on their internal utility function, which they do not have to reveal. In addition to being fair, the protocol has desirable features such as being unbiased and verifiable while allocating resources. The divide and conquer procedure whereby one agent divides the cake into two halves and the other chooses is the oldest and most elegant solution in fair division literature. Although it has many advantages, the procedure is only applicable to two agents. The attempt in this chapter is to retain the essence of divide and choose while extending it to  $n$  agents. Huhns and Malhotra [90] discussed the issue of dividing a strip of property along a coastline among 3 agents. Their procedure is an elegant solution which ensures a fair allocation of the resource. In section 4.1 I extend their solution to be applicable to any  $n$  agents. First, I present the protocol that all agents are expected to follow. Then Theorem 1 shows how an allocation is guaranteed if the agents follow the protocol. The inductive proof provides insight into how the procedure works. I prove that this protocol can enable distribution of the resource among  $n$  agents in a fair manner. Section 4.1.2 concludes with a discussion of the features of the procedure. Section 4.2 presents the case for partitioning

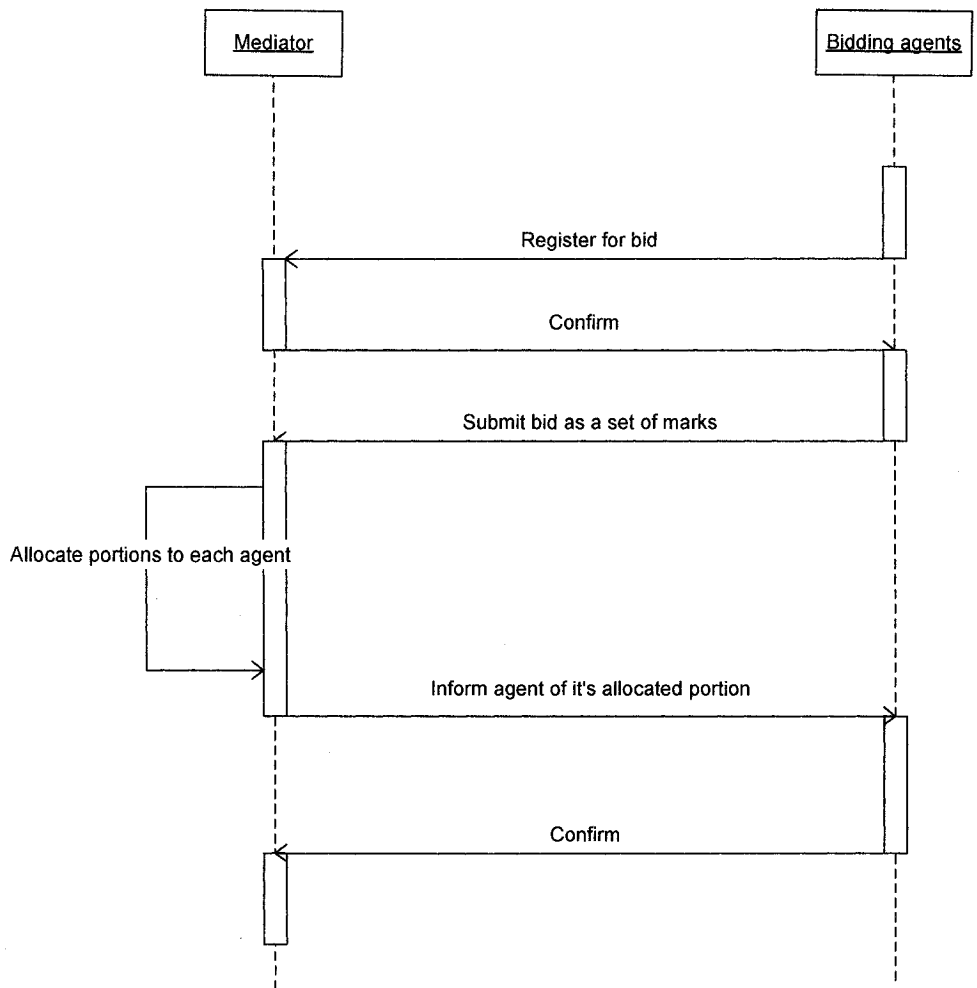
a circular resource among  $n$  agents. In this case the procedure will make radial cuts along the resource. Again, I present the circular cake cutting protocol which defines public rules for the agent to follow. Theorem 2 and its proof shows how allocation is guaranteed if agents follow the protocol. It is shown that each agent can get close to  $1/n$  of the whole resource. I end the section with a discussion of the features of the circular cake cutting procedure. The chapter concludes with a discussion of the salient features of the topological approach taken to solve the one-dimensional resource allocation problem.

## **4.1. Dividing a Linear Resource among $n$ Agents**

### ***4.1.1. Protocol***

The problem of dividing a linear resource among  $n$  agents can be solved by the following simple protocol: Each agent is required to make  $n-1$  marks of the property that delimit  $n$  intervals. To the agent making these marks, each of these intervals should be worth  $1/n$  of the whole piece and is equally desirable. The procedure guarantees that given such a set of marks, each agent will be guaranteed one of the intervals it has marked. The protocol is fair because each agent will be given one of the intervals it marked.

Refer Figure 15 to see the functioning of the protocol. Given  $n$  agents who need to divide a resource among themselves, they approach a mediator (which could be one of the bidding agent themselves) to get their portions of the resources allocated. They register with the



**Figure 15.** The linear cake cutting protocol

mediator, informing it of their interest in participating. After the mediator confirms their participation, the agents submit their bids as a set of marks denoting equal-value portions in their estimation. The mediator waits for all the agents to submit their bids. Then it calculates portions to be allocated to each agent and finally informs each agent of the portion allocated to it.

**Theorem 1.** If there are  $n$  agents and each agent makes  $n-1$  marks, creating  $n$  portions of a linear cake, then the protocol guarantees that each agent will be allotted a piece of the cake, such that the piece was one of the  $n$  portions created by the agent itself.

**Proof:** This proof assumes that the left and right portions of the cake have been allotted to the agents whose marks came first on the left side and right side respectively. We concentrate on allotting pieces of the cake to the agents still remaining after this procedure. Note that after the first 2 agents have received their share, the marks of the remaining agents need not start at the same point. Now we will have  $n-2$  agents with each agent having at least  $n-1$  marks creating at least  $n-2$  pieces of the cake. Without losing generality, we can add 2 to the above numbers and re-state the problem as: There are  $n$  agents with each agent having at least  $n+1$  marks creating at least  $n$  pieces or intervals of the cake. We prove that each agent will be guaranteed a piece of the cake such that the piece was marked by the agent himself.

When agents create marks (that represent their cuts), the following possibilities exist for a particular chosen interval.

1. Pure interval, i.e., no other agent's interval intersects this interval
2. Mixed interval
  - 2.1. The current interval intersects another interval. It does not completely contain any other interval.

2.2. The current interval contains at least one interval made by at least one agent. In addition, there will definitely be an interval that partially intersects the current interval.

**Base Case ( $n=2$ ).** Each agent will create 2 intervals, each using 3 marks. Find the first mark. Say this mark belongs to agent A. The next mark may or may not be made by A.

1. If the next mark is made by A, this is a pure interval, so allocate the current interval to A. Remove all marks to the left of this interval. Remove all marks made by agent A. We will be left with marks made by agent B to the right of the current interval. Repeat the above procedure. None of B's marks have been deleted yet. Hence the procedure is guaranteed to find an interval to allocate to B, as no other agents are left.
2. If the next mark is made by B, A's interval is mixed. Either case 2.1 or 2.2 will occur.
  - 2.1. A's interval intersects partially with B. In this case allocate the interval to A. Remove all marks to the left of interval. Remove all of A's marks. Since A's interval intersected with B's interval, the previous step will reduce B's marks to 2. Since B has two marks left demarcating an interval and no other agent is remaining, the procedure is guaranteed to find an interval for B.
  - 2.2. A contains at least one interval of B. Since there are no more agents, such an interval of B is guaranteed to be a pure interval and is allocated to B. Remove all marks to the left of the interval. Remove all of B's marks. Since A contained B, the previous step will reduce A's marks to 2. Since A has two marks left



demarcating an interval and no other agent is remaining, the procedure is guaranteed to find an interval for A.

This proves that A and B can be allocated fair shares, no matter how the marks are arranged.

**For any  $n (n > 2)$ .** Let us assume that the allocation procedure works for up to  $k$  agents.

We will show that the procedure works for  $k+1$  agents. Each agent will create  $k+1$  intervals, each, using  $k+2$  marks. Find the first mark. Say this mark belongs to agent  $i$ .

The next mark might or might not be made by  $i$ .

1. If the next mark is made by  $i$ , this is a pure interval, allocate current interval to  $i$ .

Remove all marks to the left of this interval. Remove all marks made by agent  $i$ . None of the marks made by other agents will be removed as this was a pure interval for agent  $i$ . Hence we will be left with the  $k+2$  marks and  $k+1$  intervals made by each of the  $k$  agents to the right of the current interval. Delete the leftmost mark of each of the  $k$  agents. Thus each agent is left with  $k+1$  marks and  $k$  intervals. This transforms into the allocation procedure for  $k$  agents, which we know works. Hence proved.

2. If the next mark is not made by  $i$ , suppose the mark belongs to agent  $j$ . Add  $i$  to the list of agents whose mark has already been seen. Repeat the allocation procedure with  $j$ 's mark as the first mark. At some point we will encounter a mark made by one of the agents already in the list. Say the first such agent is  $l$ . The interval demarcated by  $l$  will not contain any other agent's interval. It may or may not partially intersect with other agent's intervals.

2.1. If  $l$ 's interval does not intersect with any other interval, then allocate interval to  $l$ .

Remove all marks to the left of this interval. Remove all marks made by agent  $l$ .

The previous step will remove at most one mark of the agents. Other agents will have  $k+2$  marks and  $k+1$  intervals. Start from the beginning of the list and as each mark is encountered, check if the mark belongs to an agent already in the agent list. If so, ignore the mark; otherwise add the agent to the list and delete the mark. This step guarantees that we will have  $k$  agents, each with  $k$  intervals and  $k+1$  marks. This transforms into the allocation procedure for  $k$  agents, which we know works. Hence proved.

2.2. If  $l$ 's interval partially intersects with some other interval, then allocate the

interval to  $l$ . Remove all marks to the left of this interval. Remove all marks made by agent  $l$ . The previous step will remove at most one mark of the agents.

Other agents will have  $k+2$  marks and  $k+1$  intervals. Start from the beginning of the list and as each mark is encountered, check if the mark belongs to an agent already in the agent list. If so, ignore the mark; otherwise, add the agent to the list and delete the mark. This step guarantees that we will have  $k$  agents each with  $k$  intervals and  $k+1$  marks. This transforms into the allocation procedure for  $k$  agents, which we know works. Hence proved.

This proves that the allocation procedure works for  $n$  agents, for any  $n \geq 2$ .

#### 4.1.2. Features

If, rather, the resource is circular in shape (e.g., time slots in a 24 hour cycle or a circular cake), can one use the above procedure to get a solution? Henceforth I use the example of a circular cake to explain the division of a circular resource. A quick way to apply the linear protocol to a circular cake is the following:

The protocol begins with a mediator making a mark on the cake denoting it as a starting point. Now take a strip of length equal to the circumference of the cake. Wrap the strip around the cake such that the start and end points of the strip meet at the point marked on the cake. This will show agents the relative locations of their favored pieces. Lay out the strip on a flat surface and let  $n$  agents make  $n-1$  marks on it demarcating  $n$  intervals.

Next, use the allocation procedure for the linear problem to allocate intervals to agents. Take the strip that now shows the intervals allocated to the agents and wrap it around the cake with the start and end of the strip meeting at the point marked on the cake. Make radial portions of the cake according to the intervals marked on paper and allot them to the agents based on their markings. This protocol divides a circular cake among  $n$  agents in a fair manner. It has the following features:

1. The protocol is fair because each agent gets one of the pieces it demarcated for itself.
2. It might not be envy-free, because the initial mark made by the mediator might break up a region of interest for some agents into start and end parts. Such agents will end

up at a disadvantage over the others and might envy that the allocation procedure tilted in favor of other agents.

3. The linearization makes it difficult for the agents to visualize which portion of the cake they really wanted to have. The portions of the circular cake have to be converted to linear lengths on the strip, which can be non-trivial

Attempting to apply the linear protocol on a circular cake will end up creating a bias, because it requires a starting point to be declared, which might favor one of the agents. Is there an *unbiased* way to allocate portions of a circular cake? That is the topic of the next section.

## **4.2. A Fair and Unbiased Protocol to Partition a Circular Resource among $n$ Agents**

### **4.2.1. Protocol**

The protocol that needs to be followed by the agents is fairly simple. Given a circular cake, an agent can make radial marks at any point. If there are  $n$  agents then every agent needs to make  $n+1$  radial marks on the cake creating  $n+1$  pieces of the cake. The allocation procedure will allot one of these pieces to the agent. Thus there will be a total of  $(n+1)^2$  marks of the cake. Once agents submit their bid on how the cake will be cut, each agent will be allotted one piece of the cake demarcated by its own marks that will have a worth of  $1/(n+1)$  in its opinion. If every agent follows the protocol, then fair allotments are guaranteed for every agent.

Figure 16 shows the interaction diagram for the protocol. As will be shown later, due to inefficiency in the allocation process, there will be leftovers which may not be negligible in the valuation of the agents. In such a case the agents may ask to bid for the leftovers using the same bidding protocol.

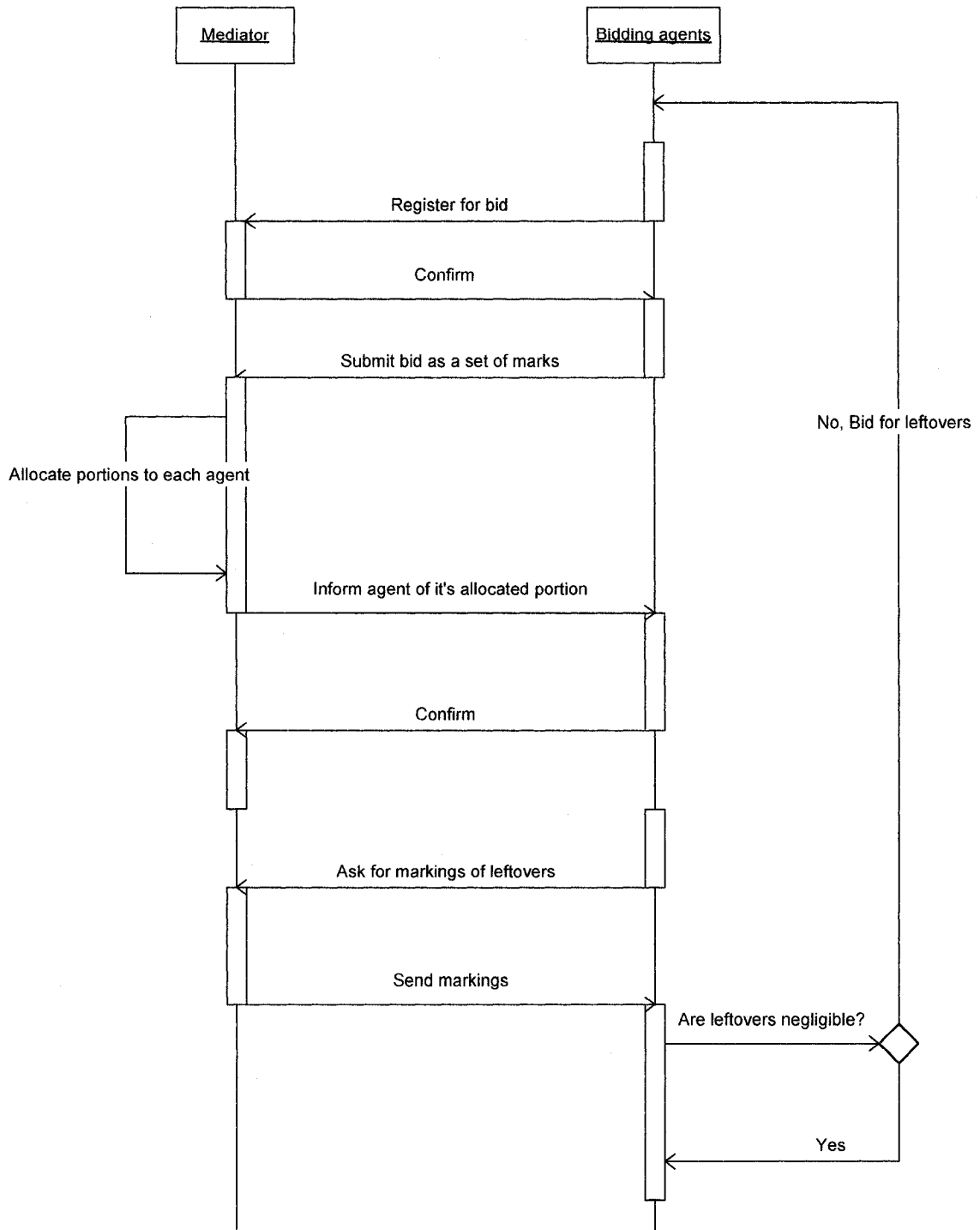


Figure 16. The circular cake cutting protocol

**Theorem 2.** If there are  $n$  agents and each agent makes  $n+1$  radial marks, creating  $n+1$  portions of the circular cake, then the protocol guarantees that each agent will be allotted a piece of the cake, such that the piece was one of the  $n+1$  pieces created by the agent itself.

**Proof:** This proof assumes that the marks are strictly in increasing order, viz. no two points are in exactly the same position. I define two terms: A  $k$ -circle and a  $k$ -sector.

**k-circle.** A  $k$ -circle consists of a circle that has  $k+1$  marks made by each of the  $k$  agents demarcating  $k+1$  intervals.

**k-sector.** A  $k$ -sector is formed when one or more intervals have been removed from a circle. A  $k$ -sector contains at least  $k+1$  marks demarcating at least  $k$  portions for each of the  $k$  agents. By this definition, a  $k$ -sector is also a  $(k-1)$ -sector, which is also a  $(k-2)$ -sector, etc. A  $k$ -sector may or may not be a  $(k+1)$ -sector however.

When agents mark their intervals, the following possibilities exist for a particular chosen interval.

1. Pure interval, i.e., no other agent's interval intersects this interval
2. Mixed interval
  - 2.1. The current interval partially intersects another interval. It does not completely contain any other interval.

2.2. The current interval contains at least one interval made by at least one agent. In addition, there will definitely be an interval that partially intersects the current interval.

In the case of a k-circle, the following events may occur:

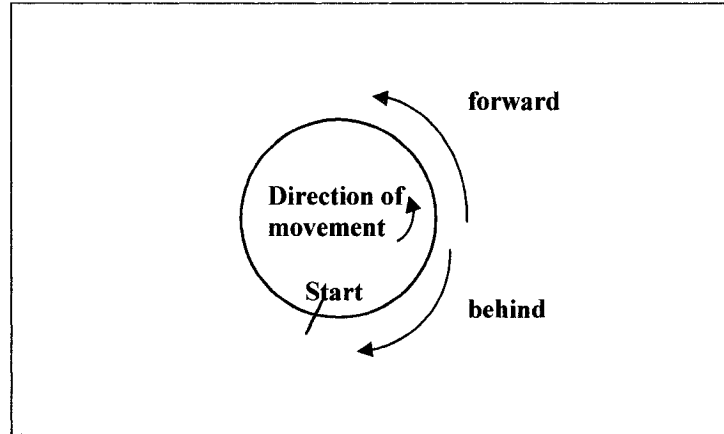
1. If the interval allotted was pure, then the other agent's intervals were outside this interval, i.e., the portions demarcated by other agents included the current interval and some extra portions ahead and behind the current interval. Since the current interval was allocated, the corresponding portions of the other agents are destroyed because they can no longer include the currently allocated portion.
2. If the interval allotted was mixed, then there was some overlap with other agents' portions. When this interval is allocated, then it destroys the two intervals of the other agents and they can no longer be allocated to the respective agents.

The above cases show that for a k-circle, allotting an interval will destroy at least one and at most 2 intervals of other agents. In the case of a k-sector, the following events may occur:

1. If the interval allotted was pure then at most 1 interval of other agents is destroyed.
2. If the interval allotted was mixed then it means other agents interval started after the current interval. Allocating the current interval would destroy at most one interval of the other agents.



The above cases show that in case of a  $k$ -sector, allotting an interval will destroy at most one interval of the other agents. I will now show an inductive proof for the allocation procedure.

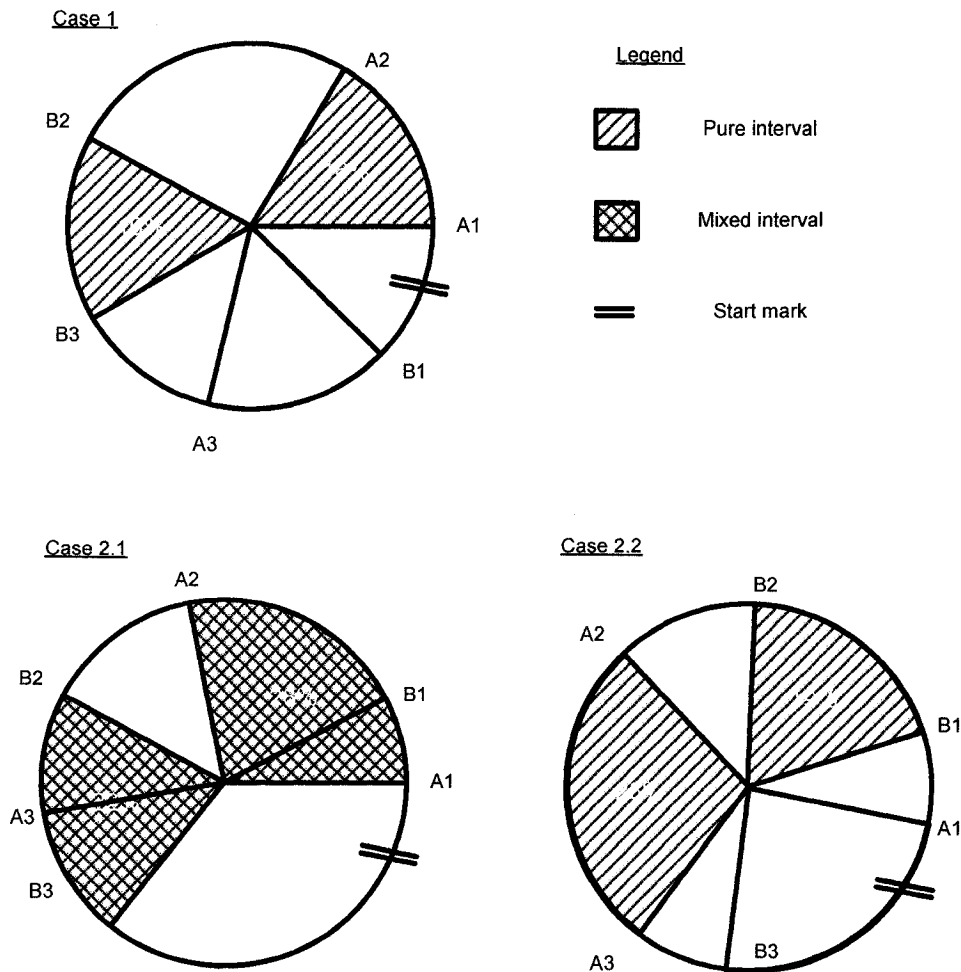


**Figure 17.** Conventions used in the proof

**Base Case ( $n=2$ ).** For 2 agents we will have a 2-circle with each agent making 3 marks and demarcating 3 intervals respectively. Let the agents be A and B. Arbitrarily mark a position on the circumference as a “start” mark. Without losing generality, we choose to move in the counterclockwise direction. In this proof, “moving back” means moving clockwise till one has reached the start mark. Similarly “moving forward” means moving counterclockwise till one has reached the start mark. Find the first mark and assume this mark belongs to agent A. The next mark may or may not be made by A.

1. If the next mark is made by A, this is a pure interval, so allocate this interval to A. Remove all marks to the back of this interval. Remove all marks made by agent A. Thus we will be left with a 1-sector having marks from agent B only. Since none of B's marks have been erased, the 1-sector will have 3 marks demarcating 2 intervals. Allot any of these intervals to B.
2. If the next mark is made by B, A's interval is mixed. Either case 2.1 or 2.2 will occur.
  - 2.1. A's interval intersects partially with B. In this case allocate the interval to A. Remove all marks behind the interval. Remove all of A's marks. We will be left with a 1-sector. Since A's interval intersected with B's interval, the previous step will reduce B's marks to 2. Thus the 1-sector will have 2 marks demarcating 1 interval. Allot this interval to B.
  - 2.2. A contains at least one interval of B. Since there are no more agents, such an interval of B is guaranteed to be a pure interval and is allocated to B. Remove all marks behind the interval. Remove all of B's marks. We will be left with a 1-sector. Since A contained B, the 1-sector will have 2 marks demarcating 1 interval. Allot this interval to A.

This proves that A and B can be allocated fair shares, no matter how the marks are arranged. Now let us turn our attention to the general case of  $n$  agents.



**Figure 18.** Allocations for  $n=2$  for a circular resource

**For any  $n (n > 2)$ .** Let us assume that the allocation procedure works for up to  $k$  agents. In order for the allocation procedure to work, one of the  $k$  agents would have been allotted an interval from a  $k$ -circle. After this a  $(k-1)$ -sector would have formed, which was used to allot intervals for the rest of the  $k-1$  agents. Hence one can conclude that the

allocation procedure works for up to  $(k-1)$ -sector. I will show that the procedure works for  $k+1$  agents.

For  $k+1$  agents create a  $(k+1)$ -circle which will have  $k+2$  marks demarcating  $k+2$  portions for each of the agents. Find the first mark. Say this mark belongs to agent  $i$ . The next mark may or may not be made by  $i$ .

1. If next mark is made by  $i$ , this is a pure interval, allocate current interval to  $i$ . Remove all marks behind this interval. Remove all marks made by agent  $i$ . None of the marks made by other agents will be removed as this was a pure interval for agent  $i$ . We will be left with a  $k$ -sector that has  $k+2$  marks demarcating  $k+1$  intervals. Read in the next mark. Say it belongs to agent  $j$ .
  - 1.1. If the mark after this belongs to  $j$  then this is a pure interval which can be allocated to  $j$ . Remove all marks made by  $j$ . We will be left with a  $(k-1)$ -sector that has  $k+2$  portions demarcating  $k+1$  intervals for each of the  $k-1$  agents. Since  $k-1$  sector problem has been solved, Hence proved.
  - 1.2. If the mark belongs to some other agent  $l$ , then add agent  $j$  to list of agents whose mark has already been seen and repeat allocation procedure for the  $k$ -sector with agent  $l$ 's mark as the first mark. Once an interval has been allocated to an agent, remove all marks of the agent. Remove all marks behind this interval. Other agents may lose at most one mark. Thus we now have a  $(k-1)$ -sector problem where  $k-1$  agents have at least  $k+1$  marks and demarcating  $k$  intervals. Hence proved.

2. If the next mark is not made by  $i$ , suppose the mark belongs to agent  $j$ . Add  $i$  to the list of agents whose mark has already been seen. Repeat the allocation procedure with  $j$ 's mark as the first mark. At some point we will encounter a mark made by one of the agents already in the list. Say the first such agent is  $l$ . The interval demarcated by  $l$  will not contain any other agent's interval. It may or may not partially intersect with other agent's intervals.

2.1. If  $l$ 's interval does not intersect with any other interval, then allocate the interval to  $l$ . Remove all marks made by agent  $l$ . We will be left with a  $k$ -sector that has  $k+2$  marks demarcating  $k+1$  intervals for each of the  $k$  agents. This  $k$ -sector is the same as the one discussed in case 1.1 whose solution has been described. Hence proved.

2.2. If  $l$ 's interval partially intersects some other interval, then allocate the interval to  $l$ . Remove all marks made by agent  $l$ . The previous step will remove at most one mark of an agent. Other agents will have  $k+2$  marks and  $k+1$  intervals. We will be left with a  $k$ -sector having at least  $k+1$  marks demarcating  $k$  intervals. Read in the next mark. Say it belongs to agent  $j$ .

2.2.1. If the mark after this belongs to  $j$  then this is a pure interval, which can be allocated to  $j$ . Remove all marks made by  $j$ . We will be left with a  $(k-1)$ -sector having at least  $k+1$  marks demarcating  $k$  intervals for each of the  $k-1$  agents. Since  $k-1$  sector problem has been solved, hence proved.

2.2.2. If the mark belongs to some other agent  $l$ , then add agent  $j$  to the list of agents whose mark has already been seen and repeat allocation procedure for the  $k$ -sector with agent  $l$ 's mark as the first mark. Once an interval has

been allocated to an agent, remove all marks of the agent. Remove all marks behind this interval. Other agents may lose at most one mark. Thus we now have a  $(k-1)$ -sector problem where  $k-1$  agents have at least  $k$  marks and demarcating  $k-1$  intervals, hence proved.

This proves that the allocation procedure works for  $n$  agents, for any  $n \geq 2$ . Next we discuss the important features of this protocol.

#### 4.2.2. Features

The protocol above has the following features:

1. The protocol is fair because each agent gets one of the pieces he demarcated for himself and that piece is worth at least  $1/(n+1)$  of the whole cake.
2. It is unbiased because there is no mediator bias as seen in the linear version of the algorithm. Each agent can have its own start mark, rather than the mediator choosing the start mark.
3. Since each agent makes  $n+1$  marks, the space complexity is  $O(n^2)$ . The protocol may make  $n$  comparisons in the first iteration,  $n-1$  comparisons in the next, and so on in the worst case scenario. Thus the time complexity is  $O(n(n+1)/2)$ .
4. One feature of this algorithm is that agents need not be ordered. Most other protocols assume that agents agree to order themselves in a certain way, but such an issue can be contentious in itself. For example, in the successive pairs algorithm agents might

prefer to come later rather than earlier to avoid the labor of redividing their portions into ever smaller pieces in each iteration.

The algorithm is inefficient, because for  $n$  agents, each agent is expected to create  $n+1$  intervals. When one of the intervals is allotted to the agent, it is actually getting  $1/(n+1)$  of the whole cake. This inefficiency can however be reduced by joining together the wasted pieces and forming a sector. All the agents can then make  $n+1$  marks demarcating  $n$  intervals in this sector. The sector algorithm can then be applied to allocate portions to each. This can be repeated until the wasted portion is negligible in every agent's opinion. The algorithm is therefore applicable only when resources are infinitely divisible into combinable portions.

Thus by reapplying the sector algorithm a finite number of times, this allocation procedure can get arbitrarily close to the  $1/n^{\text{th}}$  allocation for each agent. This protocol assumes that the problem domain is infinitely divisible and combinable. In order to give an idea of the various types of problem domains that exist, refer to Table 2.

### **4.3. Conclusion and Discussion**

I have presented a new protocol for allocating linear and circular resources, extending the classic cake cutting problem. This algorithm is applicable for  $n$  agents in general. It has been modified specially for the case of a circular resource. The proof shows that an allotment chosen by the agent itself is guaranteed for each of the  $n$  agents. In addition to

this, the protocol is fair and unbiased, features that are highly desirable but generally difficult to achieve. However, the first allotment of  $n$  pieces is inefficient, because in each agent's opinion only  $1/(n+1)^{th}$  portion of the resource is received. To improve efficiency, reapply the allotment procedure to the wasted portions of the resource. Agents can run the procedure a fixed number of times or until they all agree that the wasted portion is negligible and further division is unnecessary. The current procedure neither demands nor exposes the utility functions of individual agents. Although a mediator may be used to allot the various portions of the resource, the solution of the mediator can be verified, unlike the case of the moving knife solution. Hence disputes can be resolved easily. Also to be noted is that the mediator need not have any features, such as being unbiased. In fact the mediator can be one of the bidding agents. This is possible because the allocation procedure is completely algorithmic and does not depend on the subjectivity of the mediator. If any agent thinks the allocation was unfair, it can re-run the procedure to confirm the validity of the allocation. Unlike many other protocols, such as successive pairs or divide-and-conquer, the protocol does not require an implicit ordering of agents. This avoids any disputes as to which will be the first one to divide the resource. However, the space and time complexity of this procedure is relatively poor. Hence users will have to study the cost versus quality trade-offs to determine which protocol they find most suitable for their resource allocation needs. Among the issues that need to be looked into further are:

1. Improving the space- time complexity of the allocation procedure



2. Since the procedure requires a centralized mediator to run the allocation procedure, a distributed version of the allocation procedure could ease the load of the computation on the mediator.

# Chapter 5

## Protocols and Procedures for Two-dimensional Planar Resource Allocation

In this chapter I present a constructive solution to the classic land division problem. This result enables allocation of higher-dimensional resources without degenerating them into a one-dimensional resource allocation problem first. The allocation procedure is based on the topology of overlaps among the regions of interest of different agents. The result is an algorithm suitable for computer implementation, unlike earlier ones that were only existential in nature. It uses the notion of degree of partial overlap to create a sufficiency condition for the existence of a solution, and proposes a procedure to find the overlaps in such a case. The proposed solution is fair, strategy-proof, non-existential, and does not explicitly need the resource to be measurable. The agents do not have to reveal their private utility functions. I extend the result for one-dimensional resource allocation presented in chapter 4 to this two-dimensional one and explain the distinctive issues involved.

The protocol to be presented is *verifiable*, in the sense that the allocation of the resource is invariant to the bias of the mediator or the agent. The issue of verifiability was encountered in chapter 4, while describing the procedure for one-dimensional resource

allocation. The negotiation protocol proposed in this chapter is a one-shot protocol. This is less common than the typical iterative forms of negotiation. At the end of the negotiation, one of the agents volunteers to act as a mediator and executes the procedure. Based on the computation of agent preferences, there is one of two outcomes:

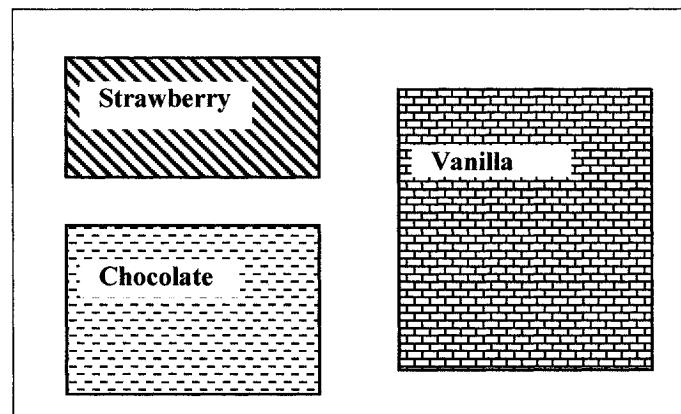
1. The procedure is able to find a solution and all agents get a fair deal.
2. The procedure is unable to find a solution and all agents receive the conflict deal, i.e., no agent receives any part of the resource.

The salient point to be noted here is that if the agent playing the role of the mediator is biased, its attempt to manipulate the results in its favor will be detected. This is because the procedure I put forth does not involve any subjective evaluations. It is an algorithmic method based on preferences that various agents have indicated. Hence the results of this method are verifiable to any agent who wants to check them. Thus the role of the mediator need not be performed by an outsider with special abilities (such as being unbiased). The resource allocation can be tackled among the agents themselves.

Another point that needs to be mentioned is that utilities of different agents are not compared. Hence utility functions of individual agents can be of any form as the procedure allocates individual portions based on the topology of the agent preferences. The problem domain of cake cutting is used as a running example to explain the procedure and discuss various related issues. In the next section, I discuss practical problems that provide the motivation for solving the two-dimensional planar resource allocation problem.

## 5.1. Motivation

### 5.1.1. Cake Cutting



**Figure 19.** Various flavors of icing on a cake

Consider a rectangular cake with three flavors of icing to be distributed among three agents as shown in Figure 19. A typical one-dimensional cake cutting procedure like moving-knife [36] can be applied to allocate portions of the cake to each of the agents in a fair manner. However If the agents have mutually exclusive strong preferences for the different icings, the allocation will not be efficient. A simple example of such a case would be when agent 1 values only the chocolate icing, while the rest of the cake is useless in its valuation. Agents 2 and 3 value the vanilla and strawberry icings respectively in a similar fashion. Efficiency in this case is meant in the pareto-optimal sense. The agents will be forced to translate their two-dimensional preferences to linear marks on the X-axis (or Y-axis), which is non-trivial. If a protocol is able to take into

account agents' preferences along both dimensions then clearly more efficient solutions can be obtained.

### 5.1.2. Land Distribution

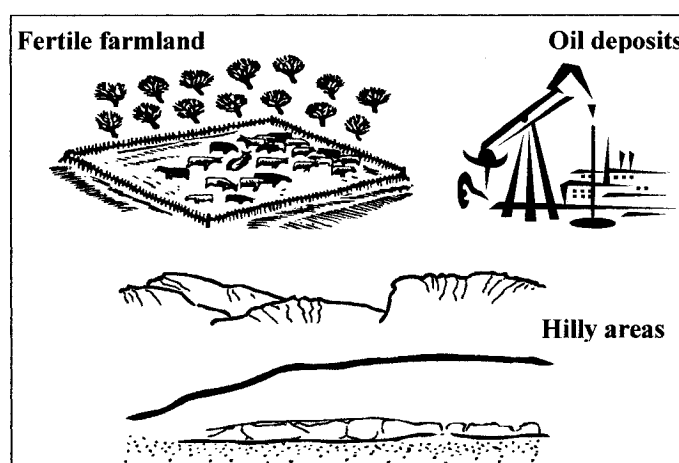


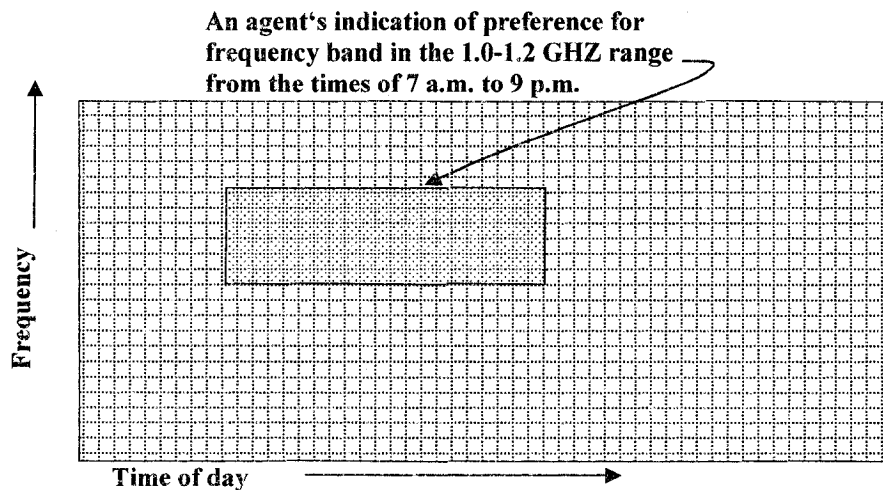
Figure 20. Land with assets.

Another problem domain that demands two-dimensional solutions is land division (). Say a tract of land has various assets like an oil field, fertile farmland and some hilly areas that need to be distributed among three agents. Each agent values the different resources differently and in order that they get the most value, it should be possible to respect their preferences in two dimensions.

### 5.1.3. Allocating Radio Spectrum

One can envision a future auction where the FCC parcels out not only frequencies but also what times of the day they are available. This will give higher efficiency, as agents

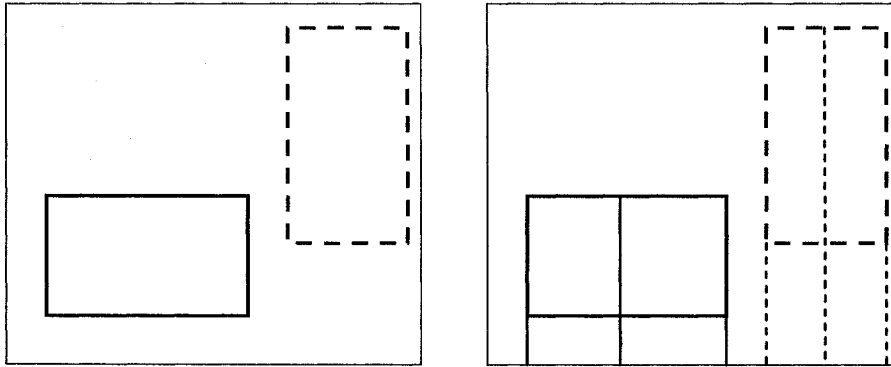
will choose the most preferable positions and the remainder can be reused by the FCC for other purposes. Thus the two independent dimensions in this scenario are time and frequency. For example, a particular radio frequency band in the 1.0-1.2 GHz range may be allotted from 7am to 9pm for broadcasting traffic information (Figure 21), while it may be used for downloading data onto wireless devices during off-peak hours.



**Figure 21.** Allocating frequency spectrum.

The examples mentioned above show that there are many real world problems whose solutions in the two-dimensional domain may prove to be far more worthwhile than using traditional one-dimensional solutions like the moving knife procedure. Essentially any real world resource, having two independent dimensions, that needs to be allocated will find the procedure I describe below to be relevant.

#### 5.1.4. Case for Higher Efficiency



**Figure 22.** a. Resources with negative utilities. b. Allocating using modified moving knife.

Now that various examples have been shown to visualize the problem space, a few points about efficiency are worth mentioning. Consider land division once again. So far I have assumed that all portions of the land have positive utility for all agents. Relaxing this assumption, it is possible that portions of land can have negative utility as well. For example, part of the land may be forest that may need to be maintained as per law, but accrue no benefit to owners. Assume that the agents view their utility as shown in Figure 22a. Agent 1 thinks that the portion allocated by the solid rectangle has positive utility, while the remaining land has negative utility. The same argument can be applied to agent 2 with respect to the dotted rectangle. Now if one uses the moving knife technique (with the knife handled by an external unbiased mediator) along one dimension, say the X-axis as per the protocol, the agent is forced to include negative utility areas as well. Hence, both agents will get very low utility (possibly negative utilities), although their allocations will be fair and envy-free. One can modify the moving knife protocol so that agents can exclude unwanted regions. But the efficiency will still be low, because the

agents cannot completely eliminate negative utility regions. Using the modified moving knife protocol the agents' divisions will appear as in Figure 22b. Despite being more efficient than the original moving knife, this modified procedure is still undesirable, because agents are forced to include negative utility areas (below the rectangles) to get their share. Obviously, if the agents were able to apportion in two dimensions, the efficiency of the overall solution could be higher. Clearly there is a need for a two-dimensional allocation procedure.

Earlier attempts to solve the two-dimensional planar resource allocation problem is found in literature, in the form of land division problems. Section 2.2.2 reviewed the literature to see how the two-dimensional cake cutting problem has been approached in the past. The next section describes the allocation procedures in detail. The features of the procedures are mentioned as well. A detailed discussion of various issues involved with the procedure is presented in chapter 7.

## **5.2. Dividing a Two-Dimensional Resource among $n$ Agents**

In this section I present the protocol that the agents must follow in order to get a fair share of a two-dimensional resource. Two different resource allocation procedures are proposed, that are applicable if agent preferences fulfill some conditions called as Iyer's conditions. If the non-overlap condition is true then the problem is mapped into a one-dimensional resource allocation problem and results from the one-dimensional resource allocation procedure is used to allocate portions to the agents. The one-dimensional



resource allocation procedure was presented in chapter 4. Since the non-overlap condition can be restrictive in how agents mark their preferences, a more flexible overlap degree condition is proposed. If the overlap degree condition is true then a novel allocation procedure is proposed, that allocates resources based on the topology of the overlaps of agent preferences. An example showing resource allocation among three competing agents illuminates the salient points of the procedure. Then a proof is presented for this procedure guaranteeing a solution for any  $n$  number of agents. The assumptions required for this procedure as well its features, are discussed in greater detail in 7.

### **5.2.1. Protocol**

The protocol proposed here is a one-shot form of negotiation. At the end of the negotiation either the procedure is able to find a solution and every agent gets a fair share of the resource, or the procedure is unable to find a solution and a *conflict deal* is reached, i.e., no agent gets any part of the resource. I describe the simple protocol below:

If there are  $n$  agents,

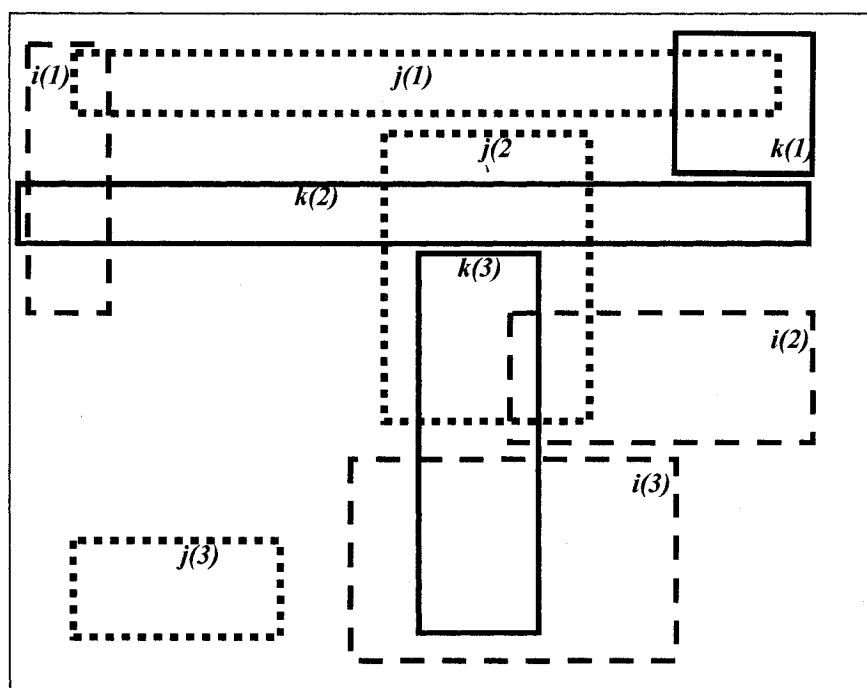
1. Then each agent will create  $n$  portions of the resource, all of which will be equal by its valuation.
2. The allocations will be in the form of rectangles. Other polygons are not allowed.
3. The portions created by a particular agent should not overlap.

If all agents have followed the protocol, then a solution is guaranteed if one of Iyer's conditions are fulfilled.

### The non-overlap condition

1. For every agent, the rectangles marked out by a particular agent do not overlap on at least one axis.
2. The above condition is fulfilled by all agents on the same axis.

I present the procedure for allocating the resource by a motivating example. Consider three agents, labeled  $i, j$ , and  $k$ , each marking three rectangles of equal value (by their own evaluation). Without losing generality, one assumes that the coordinates do not overlap on the Y-axis (non-overlap condition 2). See Figure 23.



**Figure 23.** Agents  $i, j$ , and  $k$  mark portions that fulfill the non-overlap condition. The labels of the rectangles denote their respective owners.

### 5.2.2. Procedure Given the Non-Overlap Condition

I show how the procedure works for the Y-axis, however it is applicable on any axis.

1. Project the rectangles onto the Y-axis. These will now look like closed intervals on the Y-axis. Refer Figure 24.
2. Now moving from bottom to top, extend the start point of the interval of each agent to the end point of the previous interval (belonging to the same agent).
3. The last mark (when moving from bottom to top) of every agent is extended to boundary of the resource. Refer Figure 25a. and Figure 25b.
4. One has thus transformed this problem into the one-dimensional resource allocation problem. The procedure for allocating resources in a one-dimensional case was presented in chapter 4. This procedure can be applied to obtain a fair allocation for all agents. Proceeding bottom-to-top (an arbitrary choice) and moving along the Y-axis, it is now possible to guarantee each agent will get a rectangle that belongs to itself. Theorem 1 is used to provide the guarantee that each agent gets a fair share of the resource.

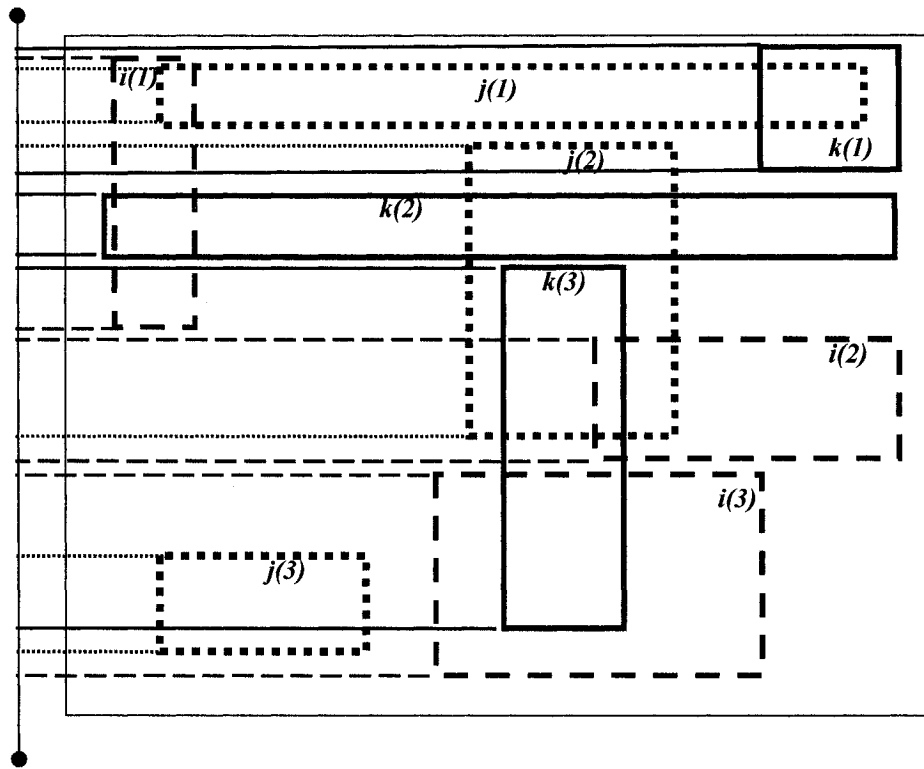


Figure 24. Projections of the agents' portions onto the Y-axis.

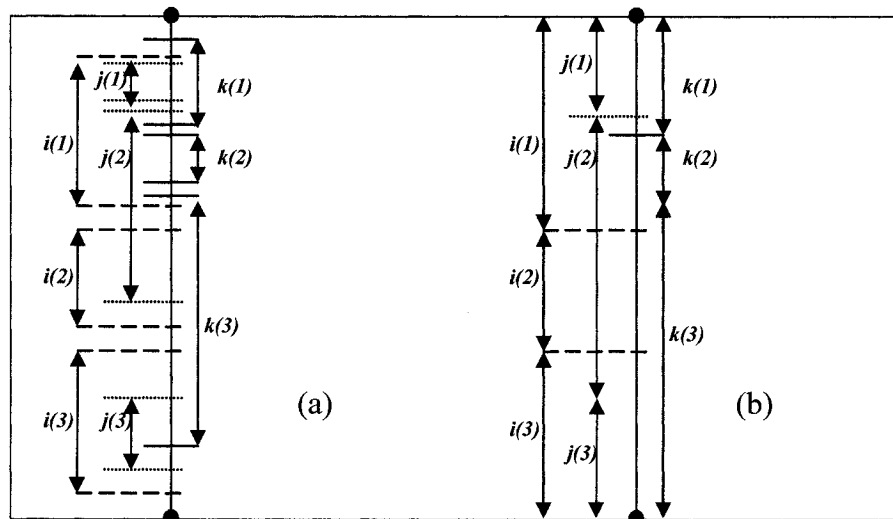


Figure 25. Transforming the two-dimensional problem into the one-dimensional case

### 5.2.3. Features of Procedure for the Non-Overlap Condition

The above procedure extends intervals. This is done in order to map the two-dimensional problem into the one-dimensional cake cutting problem. One makes use of the property that if the Y-coordinates of the rectangles made by a particular agent do not overlap, then the rectangles do not overlap at all (even if the coordinates on the other axis overlap). By extending the intervals I make sure that the whole Y-axis has been exhaustively used up. Now the problem is equivalent to the one-dimensional case, which has been tackled in chapter 4. Extending the intervals of the agents does not reduce the value of the portions allocated to them, as this is effectively like padding zero utility regions to the agents' rectangles. Who executes the procedure? As mentioned earlier, any one of the participating agents can volunteer to perform the role of a mediator and execute the procedure. If an agent (which was not the mediator) is concerned about the solution being unbiased, it can simply take the record of agent rectangles and run the procedure against them itself. The procedure is deterministic and will deliver the same results every time it is run. This is the main benefit of this allocation procedure as compared to existing ones like the moving knife method. The apportioning of resources can be *verified* by any agent who wishes to do so. Thus the role of mediation does not require external entities with special capabilities like impartiality. Due to this the agents can resolve the allocation issue among themselves which in turn protects the private information of the participants. A real example that illustrates such a scenario well is a spy satellite whose imaging services need to be allocated to different intelligence agencies. The agencies competing for the resources need not employ an external arbitrator, thereby mitigating the risk of

leaking sensitive information, while simultaneously arriving at a solution that is fair to all. Part 2 of the non-overlap condition is restrictive. In order to give more flexibility to agents to mark out their rectangles, one can modify the condition and restate it as follows:

### **The overlap Degree Condition**

Each rectangle must have a *degree of partial overlap* at most equal to one.

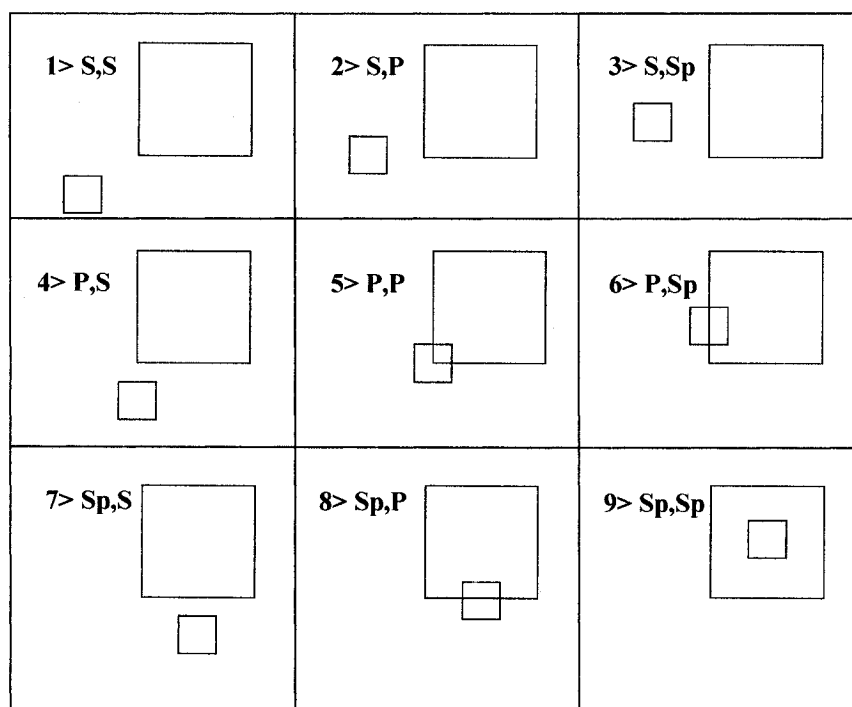
The overlap degree condition allows the agents more flexibility than the non-overlap condition. The agents can mark rectangles anywhere. Note that any two intervals belonging to a particular agent may overlap either on the X-axis or the Y-axis, but not both (If this happens then, rectangles belonging to the same agent will overlap, which violates condition 3 of the protocol). Once all the agent preferences (i.e., rectangles) are submitted to the mediator (who could be one of the participating agents), then the mediator checks that the overlap degree condition is true. If the condition is fulfilled, then the mediator can guarantee that a solution will be found.

#### **5.2.4. Degree of Partial Overlap**

What is the degree of partial overlap? I explain a few concepts before attempting to define degree of partial overlap. There are a total of 9 possibilities in which overlap can occur, between two rectangles. The types of overlap that occur along the X-axis are:

1. Separate(S): No overlap.
2. Partial(P): A partial overlap of the intervals.

3. Superset/Subset( $Sp$ ): One interval is completely enclosed in another.



**Figure 26.** The different types of overlap between two rectangles.

The same 3 cases occur along the Y-axis, thus making a total of (3 x 3) nine cases. The combinations are (S,S), (S,P), (S,Sp), (P,S), (P,P), (P,Sp), (Sp,S), (Sp,P), (Sp,Sp). Refer Figure 26. Cases 5 (P,P), 6 (S,P) and 8 (Sp, P) show the types of partial overlap that can occur between two rectangles. I mention some definitions below in order to flesh out the concept of *degree of partial overlap*.

**Definition 1:** Any two rectangles that overlap with each other are *neighbors* of each other.

**Definition 2:** Neighbors that overlap each other only partially are *partial neighbors*.

Case 9 in Figure 26 is an example of two rectangles that are neighbors of each other but *not* partial neighbors whereas cases 5,6 and 8 show examples of partial neighbors.

**Definition 3:** The *degree of partial overlap* of a rectangle is the largest number of partial neighbors it has, such that all these neighbors belong to the same agent.

Thus by verifying that the degree of partial overlap is not more than one, one ensures that not more than one rectangle of each agent overlaps partially with the rectangle under consideration. If this condition is fulfilled, then the procedure described below, is guaranteed to allocate a rectangle to each agent such that the rectangle was marked by the agent itself.

#### **5.2.5. Procedure Given the Overlap Degree Condition**

1. The rectangles are submitted to a mediator that collects them in a list.
2. The X-coordinates of the intervals are read into the Xlist. The procedure sorts the X-coordinates of the intervals in the order of their occurrence from left to right. The Xlist stores them in this sorted order. Similarly the procedure sorts the Y-coordinates of the intervals in the order of their occurrence from bottom to top. The sorted Y-coordinates are stored in the Ylist. Note that it can be determined which rectangle needs to be allotted to an agent based on the orderings of the intervals. The actual values of the X-coordinate (or Y-coordinate) of the intervals is not used for computation. Since there is no cardinal comparisons between agent rectangles (i.e., “areas” of



rectangles are not computed), one obviates the need for the resource to be measurable.

3. Determine the relation of the X intervals to each other. The following possibilities exist between any two intervals:
  - a. The intervals do not overlap (S).
  - b. The intervals partially overlap (P).
  - c. One interval is completely contained in another (Sp).

Each rectangle has a set of X relations (a relation being one of S, P, Sp or Sb) with its neighbors after one parses through the Xlist. The procedure processes the Ylist in the same way creating a set of Y relations for each rectangle and its neighbors.

	<b>S</b>	<b>P</b>	<b>Sp</b>	<b>Sb</b>
<b>S</b>	--	--	--	--
<b>P</b>	--	<b>0</b>	<b>0</b>	<b>0</b>
<b>Sp</b>	--	<b>0</b>	<b>-1</b>	<b>0</b>
<b>Sb</b>	--	<b>0</b>	<b>0</b>	<b>1</b>

**Figure 27.** The scoring matrix for a rectangle. The types of interval overlaps are Separate(S), Partial(P), Superset(Sp), Subset(Sb).

4. Create a scoring matrix. Refer Figure 27. Note that the matrix has 16 values, although I discussed nine scenarios earlier. This is because topology-wise, if interval A is the superset (Sp) of interval B, then it is the same as saying interval B is the subset (Sb) of interval A. However the scoring matrix belongs to a particular rectangle and is

from the “point of view” of a particular rectangle. Thus a given rectangle being a subset of a larger rectangle is a distinct case compared to the rectangle being a superset of a smaller rectangle. Note that the sufficient condition for any two rectangles not to overlap is that their respective intervals do not overlap in at least one of the axes. Since such rectangles are not neighbors of each other, they are not assigned a score.

5. Based on the types of overlap each rectangle has with its neighbors, one can create a directed graph that represents these connections. The directed graph is created based on rules described below. For any two rectangles:
  - a. Each rectangle is represented as a node.
  - b. If there is no overlap between the two rectangles, there is no edge between the corresponding nodes.
  - c. If there is a partial overlap between two rectangles, then each node will have a directed edge connecting the other node with an edge weight of 0.
  - d. If rectangle A is a subset of rectangle B, the edge going from Node B to Node A will have a weight of 1, whereas the edge going from Node A to Node B will have a weight of -1.
6. Once one has the topology of the connections in graph form, the procedure will arbitrarily start at some rectangle(or Node) and allot the particular rectangle based on the following conditions:
  - a. If there is no outgoing edge with edge weight of 1, then allocate this particular Node.

- b. If there is such an edge, then travel along this edge and move to the Node connected to it. Now let this be the new Node under consideration. Repeat the procedure using this node as the starting point.

Once a particular Node (or the rectangle corresponding to it) has been allocated then remove the following Nodes from the graph, for future consideration:

- c. The node just allocated.
- d. All the neighbors of the current node
- e. All the nodes belonging to the agent which was the owner of the current node

The procedure is more clearly spelled out in the following pseudocode:

```

while (graph.hasMoreNodes ())
    currentNode = graph.getNewNode ()
    while (currentNode.hasMoreEdges ())
        currentEdge=currentNode.getNewEdge ()
        if (currentEdge.getEdgeWeight ()==1)
            currentNode=currentEdge.getOtherNode (currentNode)
        end if
    end while hasMoreEdges
    currentNode.setAllocated (true)
    graph.remove (currentNode)
    graph.remove (currentNode.getNeighbors ())
    graph.remove (currentNode.getOwner ().getOwnedRectangles
    ())

```

**end while** hasMoreNodes

One point to note is that based on the rules of graph creation mentioned above, it is impossible that two nodes belonging to the same agent will ever share a common edge. This is because edges are created only between nodes whose rectangles overlap and the protocol ensures that no two rectangles belonging to the same agent overlap.

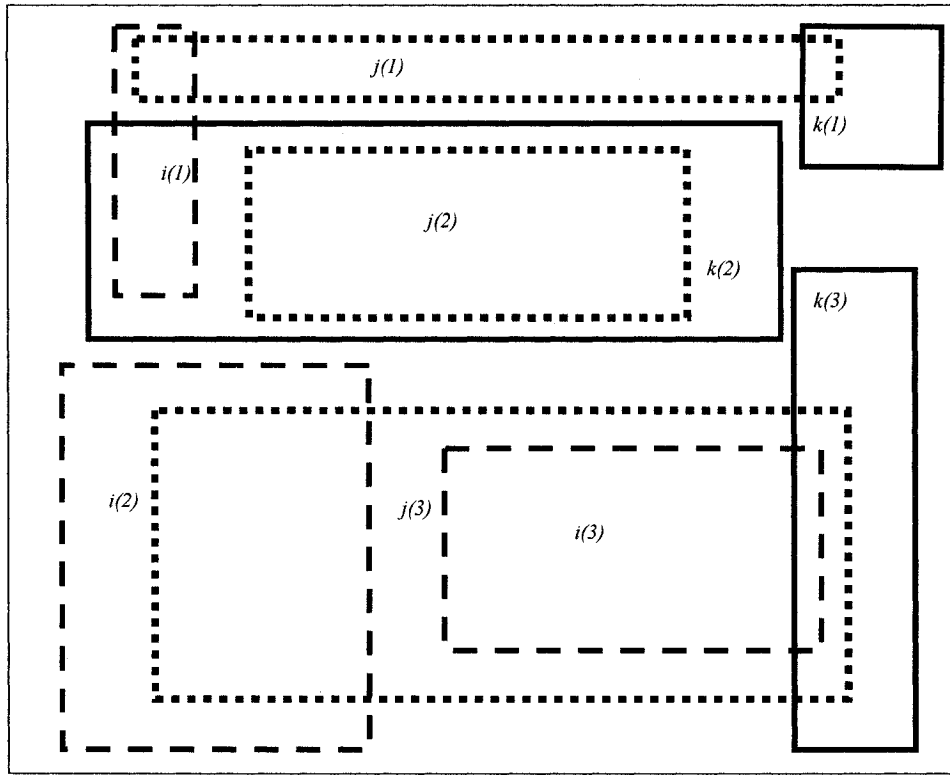
Another point is that the mediator who runs the procedure will be required to keep a list of all the Nodes traversed during procedure execution. This list will be public information accessible to all participating agents. In case an agent (which was not the mediator) needs to verify that the allocation was fair, all that it needs to do is to make sure that the path traversed by the mediator during process execution was a valid one by creating the graph of agent rectangles and executing the procedure. Note that the allocation may change if the mediator chose a different Node as a starting point. However as long as the list of Nodes traveled is made public information, the procedure is verifiable. This procedure will allocate each agent a rectangle that was marked by the agent itself and hence is a fair division procedure. I clarify the allocation procedure mentioned above with an example in the example that follows.

### ***5.2.6. An Example Allocation Problem***

I take again the running example of three agents  $i, j$ , and  $k$ , to whom a resource has to be allocated. The agents are expected to follow the protocol as laid out earlier. Each agent

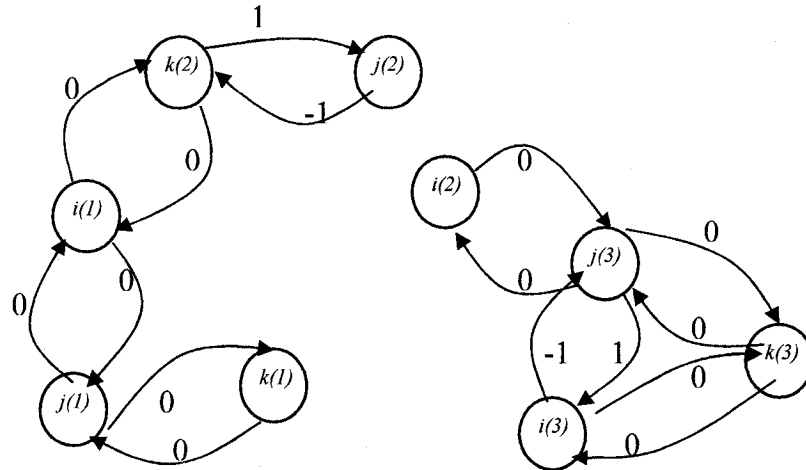
therefore marks out three rectangles on the resource, each of which it considers to be of value one-third of the total value of the resource. Based on how the rectangles are marked out, one computes that the degree of partial overlap is at most one. Hence the overlap degree condition is fulfilled. Therefore the procedure presented above should be able to find a suitable allocation. Figure 28 shows how the agents have marked out their rectangles. As described in the procedure first create a sorted Xlist and Ylist (step 2) based on the X and Y coordinates of the rectangles. Then determine how each interval overlaps with the others (step 3). Based on the scoring matrix (step 4), create a graph (step 5) representing the overlaps of the rectangles with each other. Refer Figure 29 for the corresponding graph. Next, choose any node arbitrarily (step 6), say  $i(1)$ . Observe that there are no outgoing edges from  $i(1)$  that have an edge weight of 1. So one can allot  $i(1)$  to agent  $i$ . Remove the following nodes from the graph:

- $i(1)$ : This has been allocated hence should be removed from the graph.
- $j(1)$  and  $k(2)$ : These are neighbors of  $i(1)$  and in the process of allocating node (rectangle)  $i(1)$ , nodes (rectangles)  $j(1)$  and  $k(2)$  are destroyed, hence they should be removed from the graph
- $i(2)$  and  $i(3)$ : Agent  $i$  has already been allocated  $i(1)$  and hence all of agent  $i$ 's nodes (rectangles) should be removed from the graph.



**Figure 28.** Agents  $i, j,$  and  $k$  mark portions that fulfill the overlap degree condition. The labels of the rectangles denote their respective owners.

One still has the following nodes left to be allocated:  $k(1), j(2), j(3)$  and  $k(3)$ . Select one of the nodes arbitrarily, say  $j(3)$  and apply the allocation procedure to this node (rectangle). Note that there is no outgoing edge in  $j(3)$  with edge weight 1 ( $i(3)$  has already been eliminated) . So allocate this node to  $j$ . The following nodes are removed from the graph in this process:  $j(3), k(3)$  and  $j(2)$ . Now only one node left is  $k(1)$ , which is then allocated to agent  $k$ .



**Figure 29.** The directed graph based on the topology of overlapping rectangles.

This completes the allocation procedure. If one had chosen a different node as the starting point, it is quite possible that a different set of rectangles might have been allocated to the agents. However no matter which node one selects a feasible allocation can be attained. The mediator will keep track of the nodes visited and display the information publicly in case any agents want to verify if the allocation was fair. Is there a guarantee that there exists a feasible allocation no matter how the agents lay out their rectangles? Yes, if the overlap degree condition is true. In 5.2.7 I provide proof that the procedure is guaranteed to come up with a fair allocation provided the overlap degree condition is fulfilled.

### 5.2.7. Proof given the overlap degree condition

**Theorem 3.** If there are  $n$  agents, and each agent makes  $n$  rectangles, creating  $n$  portions of a rectangular cake and if the rectangles are so marked that the degree of partial overlap is not greater than 1, then the procedure guarantees that each agent will be allotted a piece, such that the piece was one of the  $n$  portions created by the agent itself.

**Proof:**

**Base Case ( $n=2$ ).** Each agent makes two rectangles. Arbitrarily start with one rectangle. Say this rectangle belongs to agent  $i$ . Due to the non-overlap condition part 1, it is clear that none of the neighbors will belong to agent  $i$ . The following cases arise with respect to this rectangle, which is labeled as  $i(1)$ :

1.  $i(1)$  has no neighbors. In this case allocate  $i(1)$ . Remove all other rectangles made by  $i$ . None of agent  $j$ 's rectangles are destroyed in this process. The procedure is guaranteed to allocate a rectangle to  $j$ , as no other agents are left.
2.  $i(1)$  has neighbors. Note that  $i(1)$  cannot have more than one partial neighbor because at most one of  $j$ 's rectangles is allowed to intersect partially with  $i(1)$  (by condition B). The following cases arise:
  - 2.1.  $i(1)$  contains at least one of  $j$ 's rectangles completely. In this case  $j$ 's rectangle is a subset and the procedure allocates this rectangle to  $j$ . This will destroy exactly one of  $i$ 's rectangles. Remove all rectangles made by  $j$ . This leaves one of  $i$ 's rectangles intact, which can now be allocated to  $i$ .



2.2.  $i(l)$  does not contain any of  $j$ 's rectangles completely. In this case there is exactly one partial neighbor of  $i(l)$ . The procedure allocates  $i(l)$ . In this process one of  $j$ 's rectangles is destroyed. Remove all rectangles made by  $i$ . One of  $j$ 's rectangles remain which is now allocated to  $j$ .

2.3.  $i(l)$  is contained in one of  $j$ 's rectangles. This is the same as case 2.1, (by swapping the labels on the rectangles) where it has already been shown that both agents are guaranteed allocation of one of their rectangles.

This proves that  $i$  and  $j$  can be allocated their shares no matter how the rectangles are arranged  $i$  and  $j$  can be allocated their fair shares

**For any  $n$  ( $n > 2$ ).** Let us assume that the allocation procedure works for up to  $k$  agents. It will be shown that the procedure works for  $k+1$  agents. Arbitrarily choose a rectangle. Say this rectangle belongs to agent  $i$ . The following cases arise with respect to this rectangle, which is labeled as  $i(j)$  (Read as  $j^{\text{th}}$  rectangle of  $i^{\text{th}}$  agent):

1.  $i(j)$  has no neighbors. In this case allocate  $i(j)$ . Remove all other rectangles made by agent  $i$ . No other agents' rectangles are destroyed in this process. Thus one has  $k$  agents, each with  $k+1$  rectangles. Arbitrarily remove one rectangle of each of the remaining agents. Thus there are  $k$  agents, each with  $k$  rectangles which can be allocated. Hence proved.
2.  $i(j)$  has neighbors. Note that  $i(j)$  cannot have more than one partial neighbor for each of the other agents (by the overlap degree condition). The following cases arise:
  - 2.1.  $i(j)$  completely contains one (or more ) of the rectangles belonging to one (or more) of the other agents. The procedure will make the interior rectangle the

currently considered rectangle and reapply all steps beginning with case 1. In any case one of the rectangles belonging to, say, agent  $h$  will be allotted. Allocating this rectangle to  $h$ , will destroy at most one of other agents' rectangles who may be its partial neighbors (by the overlap degree condition). All agents whose rectangles formed the superset of this rectangle will lose exactly one rectangle. Next the procedure will remove the other rectangles belonging to  $h$ . Thus each of the other agents (including  $i$ ) would have lost at most one rectangle. Other agents will have  $k+1$  rectangles left. Arbitrarily remove one of the rectangles for each of the agents which have  $k+1$  rectangles. Thus there are  $k$  agents, each with  $k$  rectangles that can be allocated. Hence this is proved.

2.2.  $i(j)$  does not contain any other agents rectangles completely. Thus  $i(j)$  can have atmost one partial neighbor of each of the other agents. The procedure allocates  $i(j)$ . In this process at most one rectangle of each of the other agents is destroyed. Remove all other rectangles of  $i$ . Arbitrarily remove one rectangle of each of the agents which have  $k+1$  rectangles. Thus there are  $k$  agents, each with  $k$  rectangles which can be allocated. Hence this is proved.

2.3.  $i(j)$  is contained in some other agents' ( say one of them is agent  $h$ ) rectangles. This is the same as case 2.1 (by swapping labels on the rectangles) where it has already been shown that all agents are guaranteed allocation of one of their rectangles.

This proves that the allocation procedure works for  $n$  agents for any  $n \geq 2$ , if the degree of partial overlap is not more than one.

In the next section I discuss the various assumptions that have been made in proposing this procedure. I also discuss what parts of the solution space are covered by this procedure and the properties of such solutions.

#### ***5.2.8. Features of Procedure for the overlap degree condition***

The procedure proposed in this chapter is novel in its approach and requires further clarification about the assumptions made and a discussion of the various aspects of the algorithm. Let us begin with a discussion of the various issues involved with the proposed allocation procedure when the overlap degree condition is true .

The motivation for the procedure is the hill-climbing algorithm. The procedure tries to find the “highest point” among various nodes. If another node is on the “same level,” then moving to it does not improve our situation, hence one does not move to nodes on the same level. One also do not move to nodes on a “lower level.” If a node is on a “higher level” than the node one is currently occupying, then one has found the highest point relative to the node we started off and we allot this node (or the rectangle corresponding with this node).

The above procedure places less restriction on the way agents mark rectangles. However, the mediator will have to verify that the degree of partial overlap is not more than one, in order that it can guarantee a solution. Will the procedure work if the degree of overlap is greater than one? It is certainly possible that the procedure may find a solution, but the

theorem cannot guarantee a solution anymore and it becomes a matter of trial and error to find out the solution. If the degree of overlap is more than one, the procedure may fail for the following reasons:

- (i) A solution exists, but the procedure did not find them. A solution may have been found, had a different starting node been chosen. In such a situation, one can try each node in turn, as the first node and run the procedure to find out if an allocation exists. However, the running times may increase by an order of magnitude compared to the normal case.
- (ii) A solution does not exist. This is due to the high degree of overlap among the rectangles. In this case, the procedure will not find the solution even if one runs it repeatedly, with a different node as the starting node each time.

The analysis shows that the degree of overlap property is a *sufficient* condition for the existence of a solution, but not a *necessary* one. In chapter 7 I discuss various features of Iyer's conditions and the quality of the allocation results that they generate.

### 5.3. Conclusion

This chapter presents a constructive solution to the classic land division problem. This is the first result that is in the form of an algorithm suitable for computer implementation, unlike earlier ones (in the literature) that were only existential in nature. It uses the notion of degree of partial overlap to create a sufficiency condition for the existence of a solution, and proposes a procedure to find one in such a case. The proposed solution is fair, strategy-proof, constructive, and does not explicitly need the resource to be

measurable. It was discovered that unlike the case for one-dimensional resource allocation, there may not exist a feasible allocation for every permutation of agent rectangles in the two-dimensional case. This procedure should be taken as the first step in hyperdimensional resource allocation. While the allocation procedure is unique in the sense that it does not convert the two-dimensional resource allocation problem into a one-dimensional problem (in case of procedure given the overlap degree condition) before executing, there is nevertheless considerable scope for tweaking many parts of the procedure. Creating a distributed version of the procedure will reduce the workload on any one agent (which assumed the role of the mediator) and reduce running times. It will be interesting to see the existence and properties of procedures for higher dimensional resources and resources with topologies different from a plane.

# Chapter 6

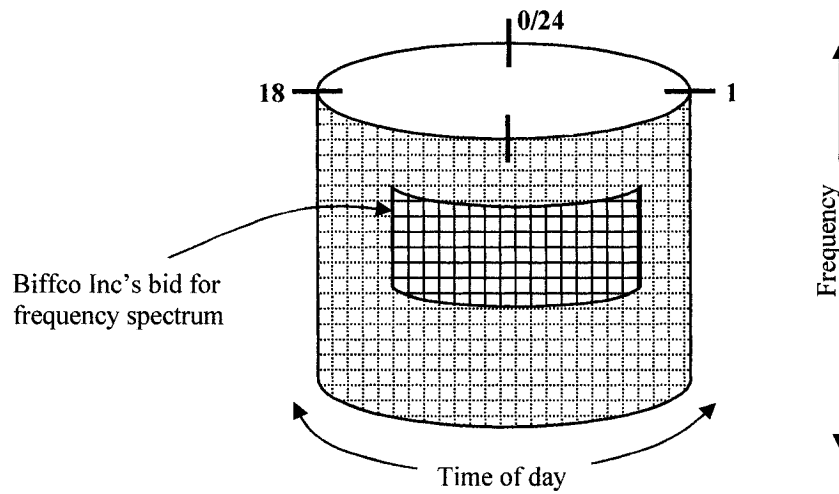
## Protocols and Procedures for Two-dimensional Cylindrical Resource Allocation

This chapter presents protocols and procedures for the allocation of resources that have two dimensions and can be characterized by the cylindrical topology. A resource has the topology of a cylindrical surface when one of its dimensions is finite and bounded (the axial dimension) and the other is finite and unbounded (herein meaning it does not have a distinct boundary). Real world examples are shown to demonstrate the practical significance of the problem. To solve it, first, it is shown how the cylindrical resource allocation problem can be mapped into a one-dimensional circular resource allocation problem for which the solution in section 4.2 applies. Second, a more flexible cylindrical resource allocation protocol is described, wherein agents mark  $n$  non-overlapping rectangles of equal value (by their valuation) on the cylindrical surface of the resource. This requires that two conditions, the non-overlap condition and the overlap degree condition be considered. If the non-overlap condition is true, then the existing allocation problem is transformed to a one-dimensional linear resource allocation problem and results in section 4.1 are applicable. The overlap degree condition uses the notion of degree of partial overlap to create a sufficiency condition for the existence of the solution. The more general two-dimensional cylindrical resource allocation procedure, to

be presented here, is then used to find the solution. The procedure is fair, strategy-proof, constructive and does not require that the resource be measurable. A discussion of the improvements that can be made to the procedure and extensions of this work to other domains and topologies is available in chapter 7. The next section provides the motivation for such problems.

## 6.1. Motivation

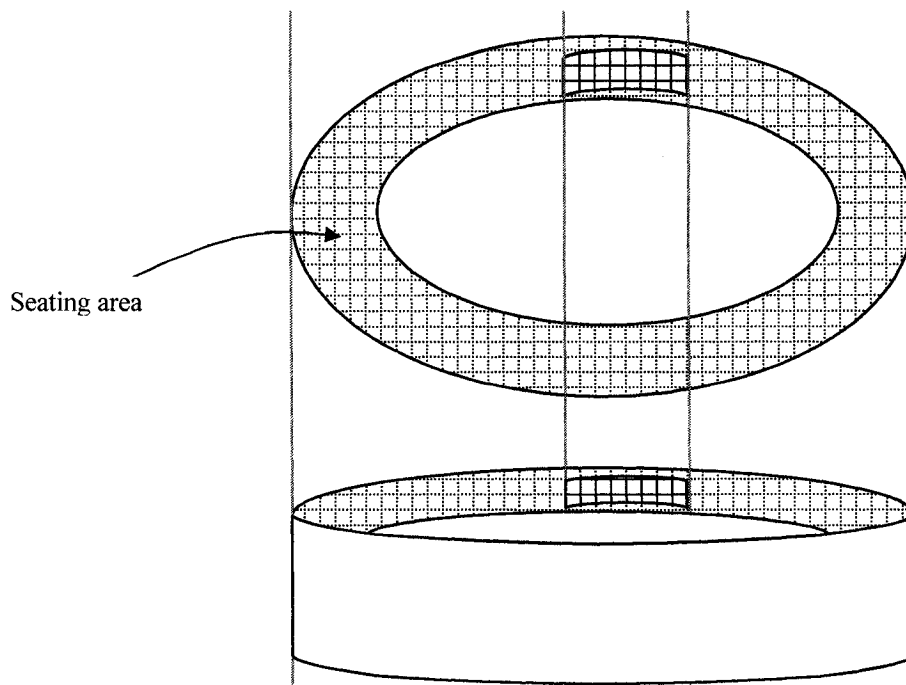
An example will illustrate the idea of a cylindrical-shaped resource. Current FCC auctions [91] for allocation of radio frequencies consists of allocating a particular band to the highest bidder. The frequency so allocated is then available to the winning agent for full-time use. Say, Acme cellular phone service wins the bid for a frequency band in the 1.0-1.2 GHz range. Due to the nature of the auction, even if the company only intends to use the allotted band from 7am to 11pm (peak usage hours); it needs to reserve the frequency for all 24 hours of the day. Another company Biffco Inc. wants to provide data streaming and downloading services to handheld devices during off-peak hours. Then it too has to bid for and reserve frequency bands for the entire 24 hours, which would be a more expensive (and potentially financially unviable) proposition. What is required is an auction mechanism where it is possible to bid for frequency bands for specific times of the day. Such an allocation makes more efficient use of the resource (i.e., the frequency spectrum). Such a modified auction would have a 24 hour clock as one dimension and frequency spectrum as the other dimension. Biffco's bid for off-peak hour usage of a frequency band is shown by the shaded rectangle in Figure 30.



**Figure 30.** Bidding for frequency bands in the modified FCC auction

Another example would be a sports stadium where seats can be auctioned off for an upcoming football match. The schematic of the sports stadium is shown in Figure 31. Various fan groups can bid for specific blocks of seats as shown by the shaded region in the figure. Since the topology of the stadium is the same as a cylinder, an allocation protocol and procedure that takes this shape into account would yield a more satisfactory result.



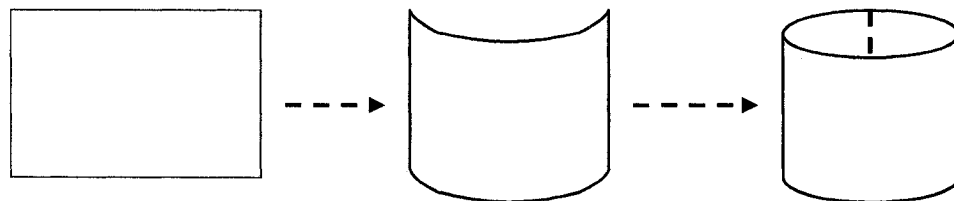


**Figure 31.** Bidding for seating blocks in a sports stadium

Can one quickly determine if a particular resource is cylindrical-shaped? Yes. A resource is said to be *cylindrical-shaped* if it is completely determined by exactly two independent finite variables, such that one of the variables is bounded while the other is unbounded. “Bounded” in this context means if the variable has distinguishable start and end points (i.e., one can identify distinct boundaries). Thus a line is a representation of a bounded variable. “Unbounded” means there exist no distinguishable start and end points. A circle (or any closed loop curve) is an example of an unbounded variable.

## 6.2. Topology of a Cylindrical Surface

In order to understand the applicability of the proposed procedure to various resource surfaces, the idea of a “cylindrical surface” needs to be explained. A cylindrical surface is obtained by gluing together a pair of opposing edges of a rectangle [92]. Figure 32 illustrates the transformation of a rectangle to a cylinder.



**Figure 32.** Transformation a two-dimensional rectangle to a cylindrical surface.




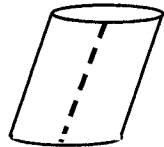

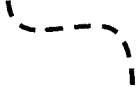

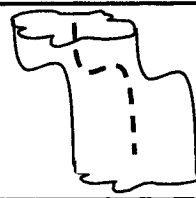




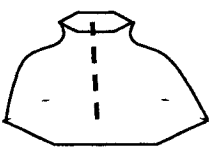



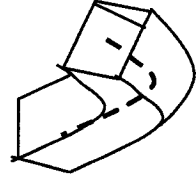




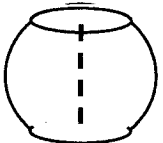



A cylindrical surface will have two surfaces and two edges. Shapes with very different geometry can have the same topology. A flat disc with a hole cut in the middle is actually topologically equivalent to the cylinder surface because it has the same number of edges and surfaces as that of a cylinder. The procedure and protocol that is proposed in here is *not yet* applicable to any shape of the cylindrical surface. Hence I specify the constrained set of shapes to which the procedures are applicable. The procedure is applicable to any cylindrical surface generated with the following parameters (qualified by constraints):

1. Shape of the generating curve: The generating curve should be closed loop and planar. Thus circles, ellipses or any arbitrary amoeba shapes on a two-dimensional

plane qualify. The shape of the generating curve should not change as it moves along the axis, although the size can change.

2. Shape of the central axis: This is the line along which the curve moves to generate the cylindrical surface. The axis should be of finite length and can be any of any arbitrary shape in three dimensions as long as it does not form a closed loop.
3. The orientation of the generating curve with respect to the axis: The plane of the generating curve can be oriented at any *constant* angle with respect to the axis. However it cannot be parallel to the axis (as this would not generate a cylinder). Thus it is required that the angle of orientation be a non-zero finite value.
4. The scaling function for the generating curve: This parameter determines if the generating curve expands or contracts as it moves along the axis. This is used to generate cone and pot shaped surfaces. The scaling function can vary along the length of the axis, but it should always have a non-zero finite value.

Thus a cylindrical surface is generated by moving a planar closed-loop curve (of arbitrary shape) - oriented at a certain fixed angle with respect to the axis – along the axis (which is not a closed loop) while varying in size according to the scaling function.

Shape of generating curve	Shape of central axis	Orientation	Scaling function	Shape of cylinder
			None	
			None	
				
			None	
				 Sphere with holes in the top and bottom
	Single Point		Constant value	 Flat disc with hole in the middle

**Figure 33.** Generating cylindrical-shaped surfaces based on the various values of parameters

The parameters and their respective constraints serve to describe the widest variety of shapes that are applicable for the procedure while ensuring that topological limitations

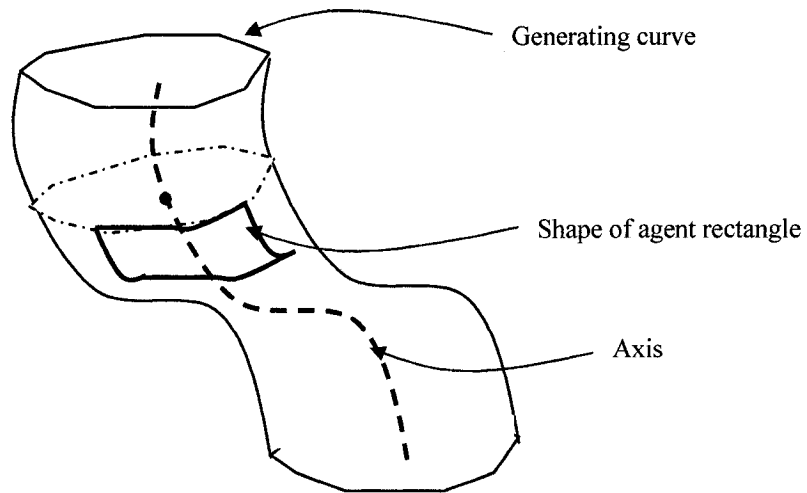
are respected. Most of the standard cylindrical shapes that are encountered in real life are covered by the above description.

Figure 33 shows some examples of well known cylindrical shapes along with the values for parameters needed to generate them. Note that the description still leaves out some shapes that have the topology of a cylinder. These are shapes generated by the planar curve that have varying angle of rotation as it moves along the length of the axis. The proposed procedure does not apply for such shapes.

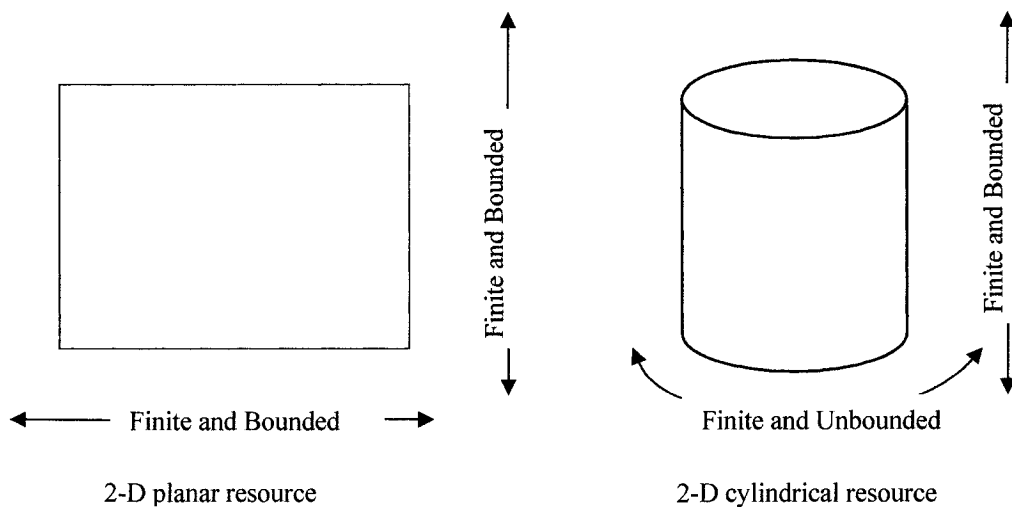
The protocol to be put forth in the next section requires agents to draw rectangles on the cylindrical surface. How does one draw a rectangle on applicable cylindrical surfaces? I follow the analogy of drawing a two-dimensional planar rectangle where each pair of opposing sides is parallel to the X-axis and Y-axis respectively. In case of a cylindrical surface, one pair of sides is drawn parallel to the central axis (which need not be a straight line). The other pair of edges is drawn collinear to the generating curve at that point of the axis. Once rectangles are drawn in this manner (Figure 34), the proposed procedure can be applied to find feasible allocations. For ease of visualization, however, I will explain the details of the protocol and procedure using the standard geometric cylinder as a stand-in for all the applicable types of cylindrical surfaces.

A protocol and procedure for the allocation of a two-dimensional planar resource has been described in chapter 5.2. In this chapter, protocols and procedures are proposed for the allocation of two-dimensional cylindrical resources that applies some of the ideas of

the earlier work. The topological difference between a planar and cylindrical resource is shown in Figure 35.



**Figure 34.** Drawing a rectangle on a cylindrical surface



**Figure 35.** Topologies of planar and cylindrical resources

It is important to note that only the topology of the surface is important, not the geometry. Thus any surface having the topology of the cylinder is part of the domain of the proposed allocation protocol and procedure. I model the resources as a standard geometric cylinder simply because it is easier to illustrate the concepts, but the results apply to any shape that can be obtained from the continuous transformation of the cylindrical surface.

The literature survey shows that there has been no explicit discussion on the allocation of cylindrical-shaped resources. This is because there are some implicit assumptions made about the "shape" of the resource, i.e., the resources have been assumed to be one-dimensional and open-ended. The resource to be divided is commonly taken in the shape of a rectangular cake, wherein a knife is held parallel to one of the edges and cuts portions of the cake to allocate to agents. Work that is relevant to this chapter exists in the form of discussion on allocation of land (modeled as a two-dimensional resource). The relevant literature on land division is presented in section 2.2.2. In the next section, I describe the details of the cylindrical resource sharing approach.

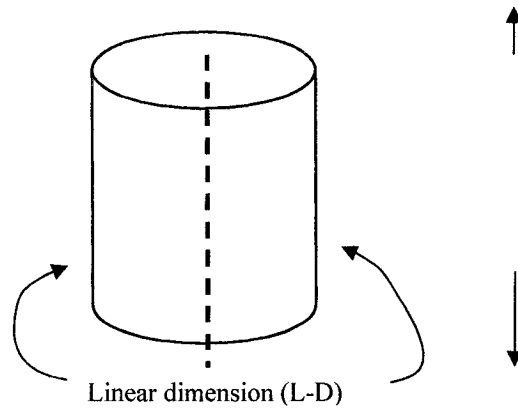
### **6.3. Dividing a Two-Dimensional Cylindrical Resource among $n$ Agents**

Work in chapter 4 presented solutions for one-dimensional linear RA and one-dimensional circular RA problems. Based on how the protocols and procedures are set up, one can exploit those results for the cylindrical RA problem. Section 6.3.1 presents the framework for transforming the cylindrical RA to circular RA problem. Then in

section 6.3.2 a less restrictive protocol and procedures are proposed that make use of results in the linear RA and planar RA research to come up with an expanded space of solutions. The protocol is a simple one-shot type of protocol. At the end of the negotiation either the procedure is able to find a solution and every agent gets a fair share of the resource, or the procedure is unable to find a solution and a conflict deal is reached i.e., no agent gets any part of the resource. After the agents have indicated their preferences, a mediator will run the procedure to allocate the resources. Who is the mediator? The mediator can be chosen from among the participating agents themselves. Because the procedure is verifiable (i.e., any agent who wants to double check the allocation process can simply run the procedure again after obtaining preference information of other agents), a mediator does not need to possess special qualifications like being unbiased for example. Thus mediator bias does not become a factor because the allocation procedure is deterministic. As the mediator is chosen from among the agents, information of agent preferences is not revealed to external parties and better privacy is afforded.

A cylindrical resource has two distinct dimensions. One dimension is parallel to the axis of the cylinder and is finite and bounded. This is referred to as the Linear dimension (L-D). The other dimension is perpendicular to the axis of the cylinder and runs along the circumference of the circle that forms the cross-section of the cylinder. This dimension is finite and unbounded. This is referred to as the Circular dimension (C-D). Refer Figure 36.





**Figure 36.** The dimensions of a cylindrical resource.

### **6.3.1. Mapping the Cylindrical RA Problem to the Circular RA Problem**

The following protocol is a modification of the protocol specified for the circular RA problem in section 4.2.1.

#### **Circular RA Protocol**

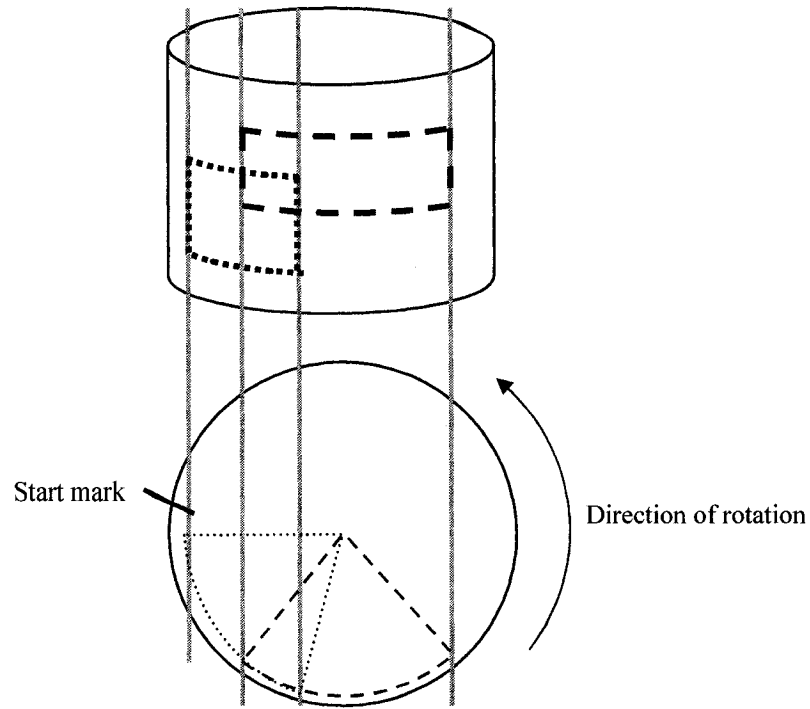
If there are  $n$  agents,

1. Each agent will create  $n+1$  portions of the resource all of which will be equal by its valuation.
2. Agents can create portions only by marking rectangles on the cylindrical surface. Other shapes not allowed.
3. Portions created by the same agent should not overlap with each other along the C-D.

Thus  $n$  agents will demarcate  $n+1$  non overlapping (along the C-D) rectangles creating a total of  $n(n+1)$  rectangles. If every agent follows the protocol then fair allotments are guaranteed for every agent.

### **Circular RA Procedure**

1. Project the rectangles onto the C-D. This will look like closed intervals on the C-D. Refer Figure 37.
2. Choose a start mark on the C-D and move in the counterclockwise direction (both chosen arbitrarily).
3. Extend the end point of each interval till the start point of the next interval belonging to the same agent.
4. This problem is thus transformed into a circular RA problem. The procedure for allocating resources was presented section 4.2. This procedure can be applied to obtain a fair allocation for all agents.



**Figure 37.** Projection of rectangles onto the circular dimension.

The following theorem guarantees that each agent will receive a fair share that of the resource.

**Theorem 2.** If there are  $n$  agents and each agent makes  $n+1$  radial marks, creating  $n+1$  portions of a circular resource, then the procedure guarantees that each agent will be allotted a portion of the resource, such that the portion is one of the  $n+1$  portions created by the agent itself.

The proof and the procedure for this theorem are described section 4.2.

## Features

The procedure is fair in the sense that if agents adhere to the protocol, then each agent is guaranteed a portion of the resource which was demarcated by the agent itself. However since each agent creates  $n+1$  equal portions of the resource, the portion it receives will be worth at most  $1/(n+1)$  of the entire resource. This makes it inefficient in comparison to other procedures which strive to give each agent at least  $1/n$  of the entire resource. However the advantage of this procedure is that no mediator bias is involved as the procedure is completely deterministic and can be verified by any agent by running the procedure again.

### 6.3.2. Mapping the Cylindrical RA Problem to the Linear RA Problem

In this section, I present a protocol that enables the mapping of the cylindrical RA into the linear RA. It is called the *cylindrical RA protocol* because the same protocol can be used for the native two-dimensional cylindrical RA procedure that will be presented later.

#### Cylindrical RA Protocol

If there are  $n$  agents,

1. Then each agent will create  $n$  portions of the resource, all of which will be equal by its valuation.
2. Agents can create portions only by marking rectangles on the cylindrical surface. Other shapes are not allowed.

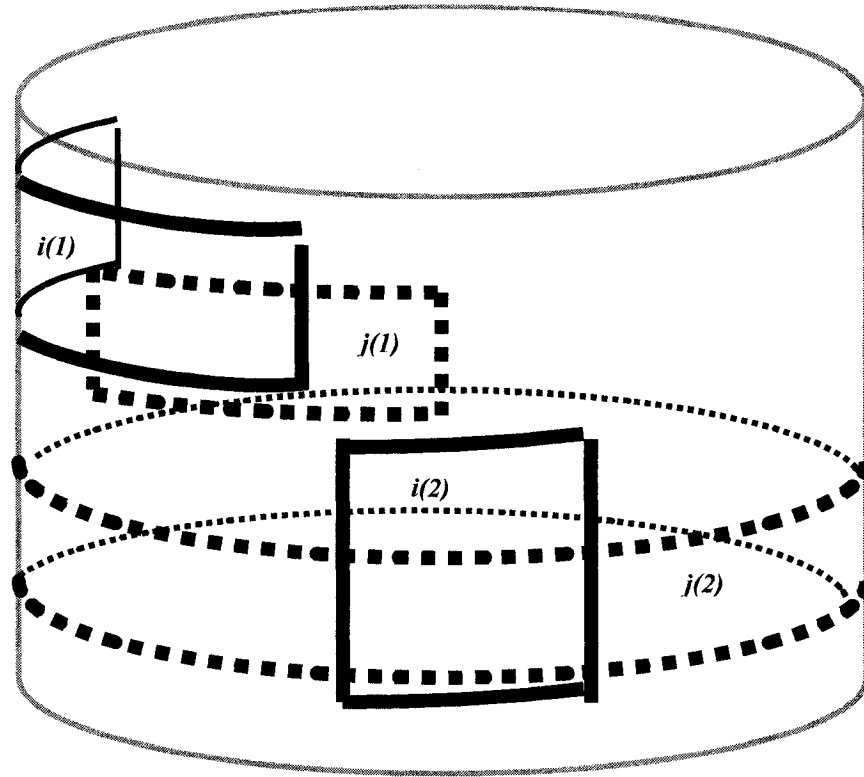
3. Portions created by the same agent should not overlap with each other.

In order to guarantee an allocation, it has to be verified that agent preferences fulfill some qualifying conditions.

### **The non-overlap condition**

For every agent, the rectangles marked out by a particular agent do not overlap along the L-D.

If the non-overlap condition is true then, one can map the two-dimensional cylindrical resource allocation problem (cylindrical RA) into a one-dimensional linear resource allocation problem (linear RA). Procedures for solving a linear RA problem was proposed in section 4.1 and can be applied in this case. The procedure for allocating the resource is presented by the following example allocation problem. Consider two agents, labeled  $i$  and  $j$  each marking two rectangles of equal value (as per the requirement of the protocol). See Figure 38.



**Figure 38.** Agents  $i$  and  $j$  mark out two rectangles each on the cylindrical surface. Agent  $i$ 's rectangles are represented by solid lines, while agent  $j$ 's rectangles are marked by the dotted lines.

### **Procedure given the non-overlap condition**

Due to the non-overlap condition, it is known that rectangles belonging to a particular agent do not overlap along the L-axis. The following procedure can then be applied.

1. Project the rectangles along the L-axis. These will now look like closed intervals on this axis (see Figure 39).
2. Starting from the bottom and moving to the top, extend the start point of the interval of each agent to the end point of the previous interval (belonging to the same agent)

3. The last mark (when moving from bottom to top) of every agent is extended to the boundary of the resource (see Figure 40).
4. One has thus transformed the cylindrical RA problem into a linear RA problem. The procedure for allocating resources was presented in section 4.1. This procedure can be applied to obtain a fair allocation for all agents. Proceeding from bottom to top (an arbitrary choice) and moving along the L-axis, it is now possible to guarantee each agent will get a rectangle that belongs to it.

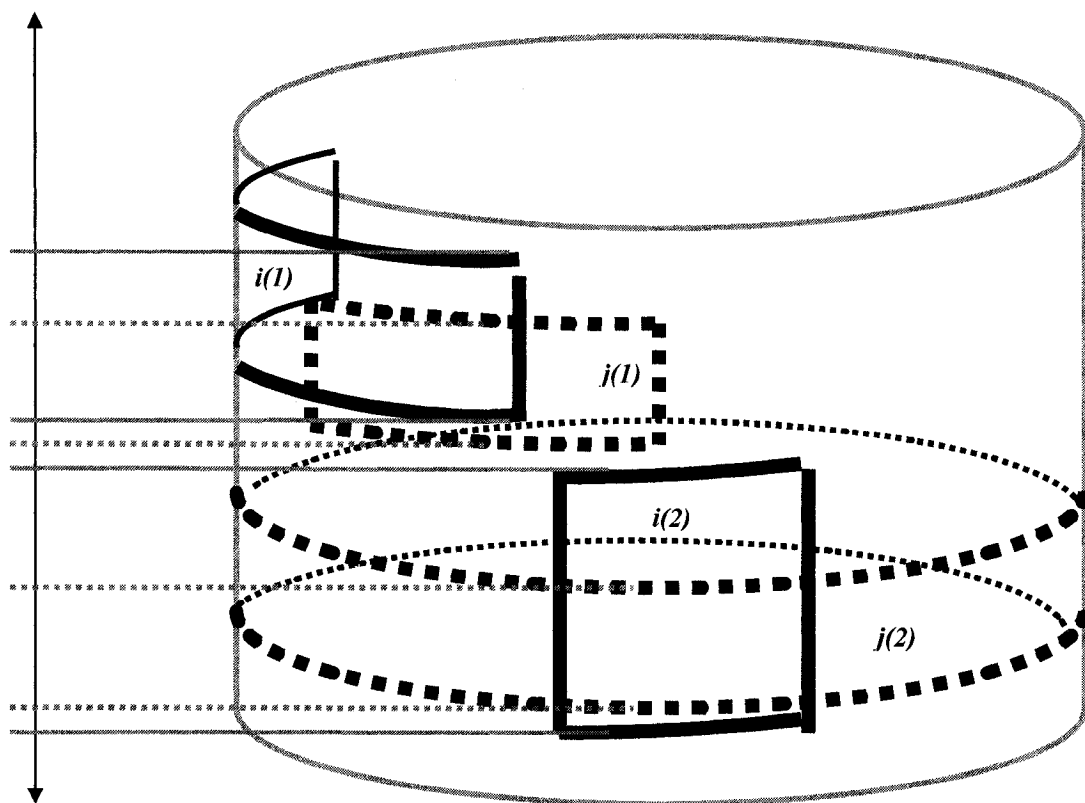
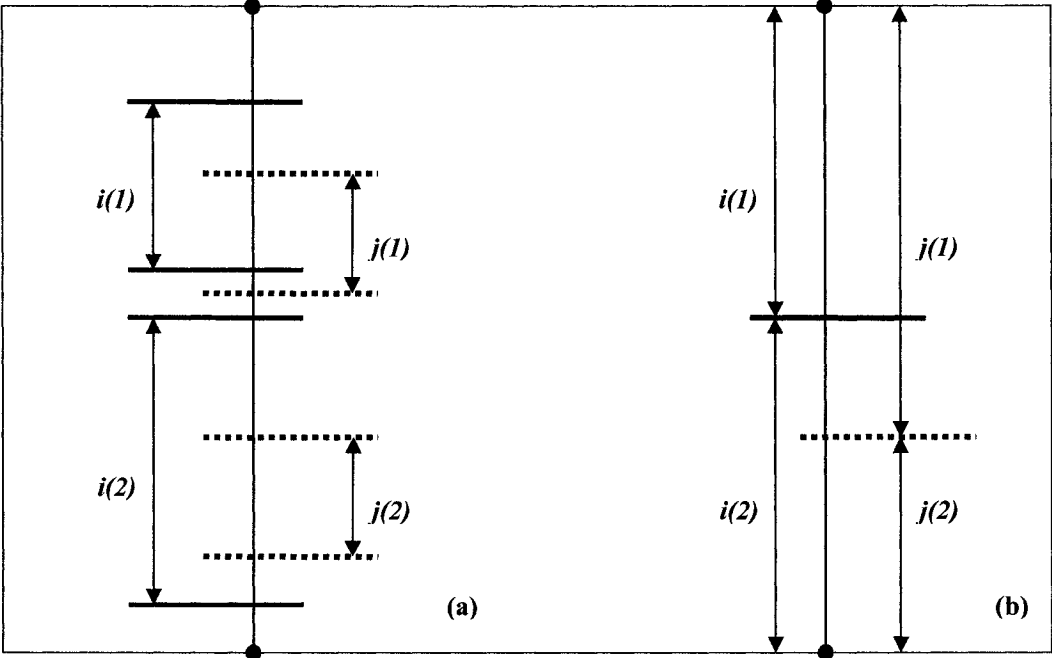


Figure 39. Projections of the agents' portions onto the L-axis

The following theorem guarantees that each agent will receive a fair share that of the resource.

**Theorem 1.** If there are  $n$  agents and each agent makes  $n-1$  marks, creating  $n$  portions of a linear resource, then the procedure guarantees that each agent will be allotted a portion of the resource, such that the portion is one of the  $n$  portions created by the agent itself.

The proof and the procedure for this theorem is described in section 4.1.1.



**Figure 40.** Transforming the two-dimensional cylindrical RA into the one-dimensional linear RA



## **Features of Procedure for the non-overlap condition**

The procedure for the non-overlap condition modifies agent intervals. This is done in order to map the cylindrical RA problem to the linear RA problem. However this modification is benign because the modified intervals are effectively the original intervals padded with zero utility regions. Once the allocation procedure terminates, one can easily obtain the original intervals by removing the padded regions (in case the padded region is of negative utility to the agent). I exploit the property that if the co-ordinates of any two rectangles made by a particular agent do not overlap along the L-axis, then the rectangles do not overlap at all (even if the co-ordinates overlap along the C-axis). By extending the intervals along the L-axis, one ensures that entire L-axis is exhaustively used up. Now the problem is equivalent to the linear RA problem which has been tackled in chapter 4

### ***6.3.3. A Native Cylindrical RA Procedure***

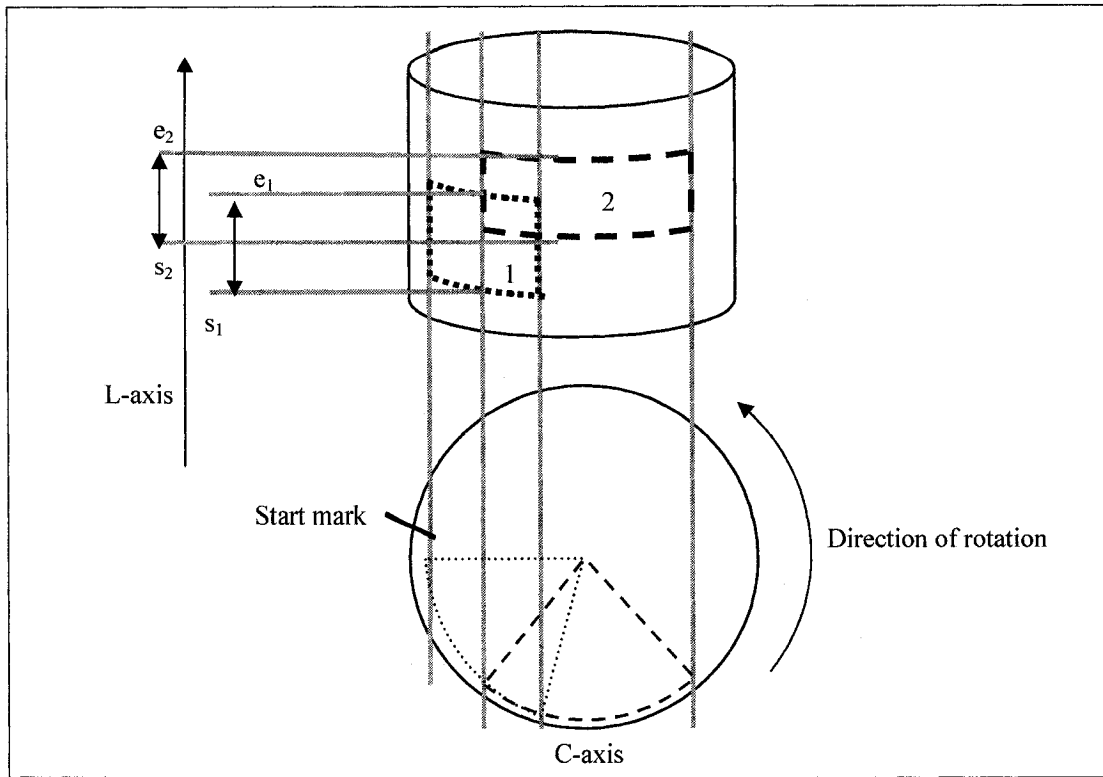
The protocol required for the native cylindrical RA procedure has already been mentioned in the previous section. The non-overlap condition is restrictive on how agents should mark their rectangles. The overlap degree condition is put forth to allow agents greater flexibility in how agents should draw rectangles on the surface of the cylindrical resource while still guaranteeing an allocation can exist. If none of the above conditions hold true then the procedure (for the overlap degree condition) can still be run to check for the presence of a feasible allocation but no guarantees can be made. The procedure is novel because it takes into account the topology of the overlaps of agent preferences on a

cylindrical surface. An example showing resource allocation among three competing agents gives an intuition of the procedure. This is followed by proof that the procedure is guaranteed to find an allocation when the overlap degree condition is true. I begin by stating the overlap degree condition:

### **The overlap degree condition**

Each rectangle must have a degree of partial overlap at most equal to one.

The overlap degree condition permits greater flexibility in how agents can mark their rectangles. As per the cylindrical RA protocol, it is still required that rectangles belonging to the same agent do not overlap. The sufficient condition for this is that the corresponding intervals along the L-axis or the C-axis do not overlap. Note that the non-overlap condition was more rigid because the overlaps of intervals were to be avoided specifically on the L-axis. Agents specify their preferences (by marking out rectangles) to a mediator. The mediator then determines whether the overlap degree condition is fulfilled. If so, then a solution can be guaranteed and the allocation procedure is executed.



**Figure 41.** Nature of overlap of agent rectangles along the C-axis and the L-axis

The notion of the degree of partial overlap was explained in section 5.2.4. It is elaborated here in the context of a cylindrical topology. The notion of overlap is explained first. Figure 41 shows two rectangles drawn on the surface of the cylinder that overlap each other. The corresponding intervals are projected on to the L-axis and the C-axis. The following types of overlap (relationship), between any two intervals, can occur along each axis:

1. Separate(S): The intervals do not overlap.
2. Partial (P): There exists a partial overlap between the intervals.
3. Superset/Subset (Sp/Sb): One interval is completely enclosed by the other.

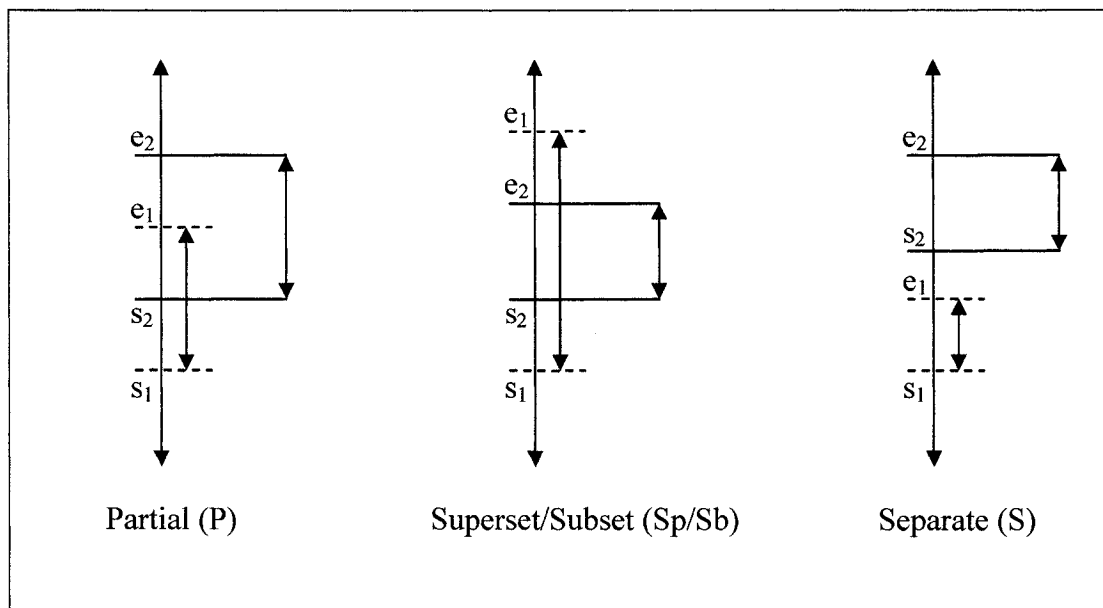
The relationship between any two rectangles can be completely determined by the relationship between the corresponding intervals along the L-axis and C-axis respectively. The location of a rectangle is completely specified by two values along the L-axis (L-points) that form the L-interval and the two values along the C-axis (C-points) that form the C-interval. Hence once the procedure has a list of L-interval and C-interval positions, it should be able to determine the relationship between any two rectangles. Given any two pairs of L-interval points, it is possible to find the relationship between the corresponding L-intervals. I choose an arbitrary convention of moving from bottom to top along the L-axis, reading off L-points I go. The start point of an interval is denoted as  $s$  and the end is denoted as  $e$ . The subscript denotes the label of the rectangle that is specified by the points. Figure 42 shows why the  $s_1- s_2- e_1- e_2$  combination of points is denoted as a Partial overlap between intervals. The table below shows the various permutations of points and the corresponding relationship between the intervals. Topologically only a few of the combination of points are unique. I exploit the following property to weed out redundancies:

### **Swap invariance property**

The relationship between intervals is invariant with respect to the swapping of their respective start labels and end labels.

	Ordering	Relationship
1	$s_1- s_2- e_1- e_2$	P
2	$s_1- s_2- e_2- e_1$	Sp/Sb
3	$s_1- e_1- s_2- e_2$	S
4	$s_2- s_1- e_1- e_2$	Sp/Sb
5	$s_2- s_1- e_2- e_1$	P
6	$s_2- e_2- s_1- e_1$	S

**Table 3.** Relationships between two L-intervals based on the ordering of the points



**Figure 42.** The topologically unique overlaps of L-intervals

Thus  $s_1 \leftrightarrow s_2$  and  $e_1 \leftrightarrow e_2$  does not affect the relationship between intervals. For example, if one swap start labels and end labels in case 1:  $s_1- s_2- e_1- e_2$  becomes  $s_2- s_1- e_2- e_1$  which is case 5 (a partial overlap). The swapping property holds for any of the above combination of points. This symmetry reduces the number of combinations of interest to 3 unique types. Cases 1, 2 and 3 are the only topologically unique combinations that

exist. They map to the three type of overlaps P, Sp/Sb and S respectively and are shown in Figure 42.

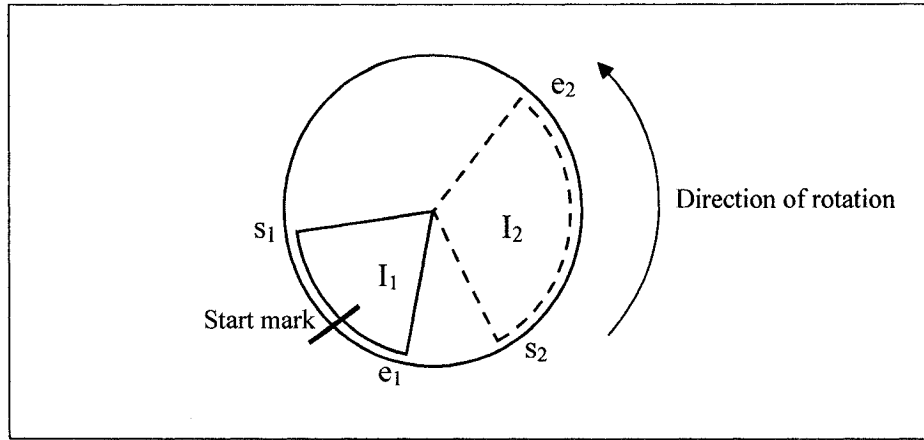
The relationship of the intervals along the C-axis is bit more complicated. The following conventions are used:

- One moves in the counterclockwise direction while recording marks.
- One marks an arbitrary position on the circumference as the start mark.

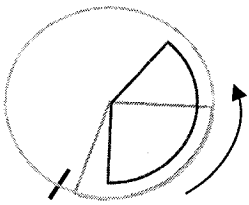
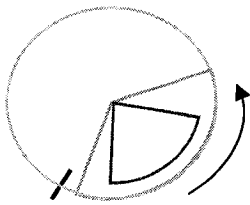
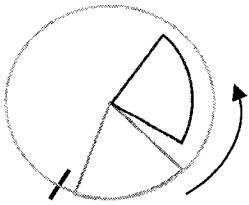
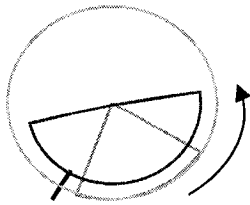
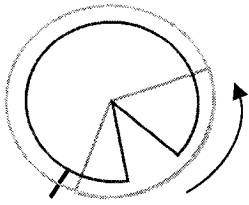
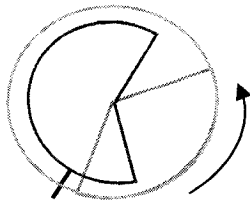
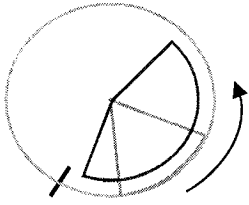
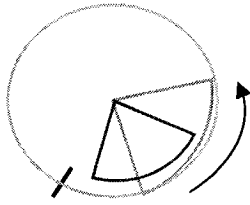
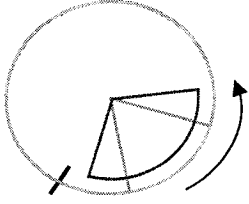
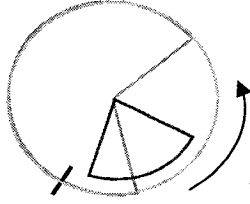
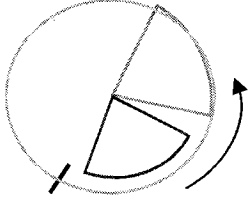
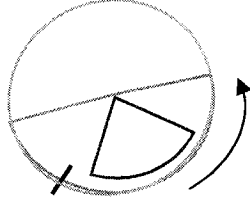
Note that in the case of a linear dimension (L-axis) the end point( $e_1$ ) of an interval would not be encountered before the start point( $s_1$ ). But because a closed loop (C-axis) is topologically distinct from a linear dimension, end points can occur before start points.

Figure 43 shows an example where  $e_1$  is read in before  $s_1$  and how the interval is to be interpreted. The 24 combinations of  $s_1, e_1, s_2, e_2$  are described in Table 4 along with the relationships they imply.

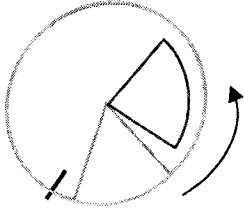
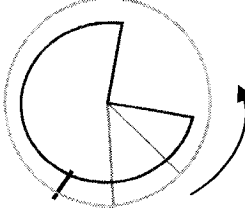
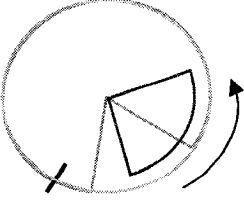
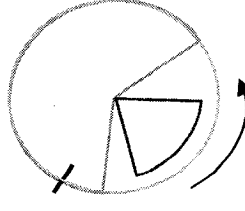
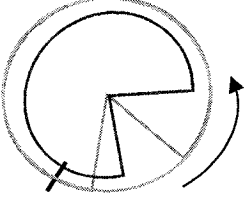
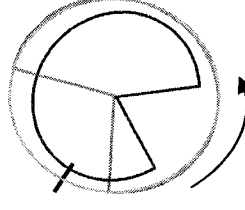
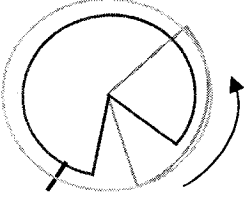
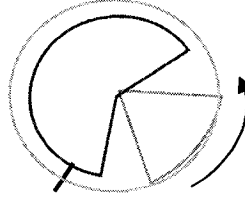
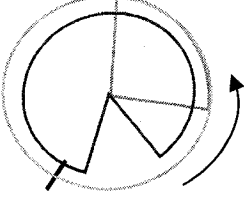
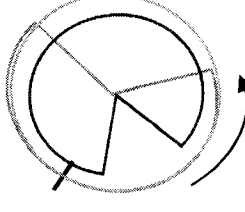
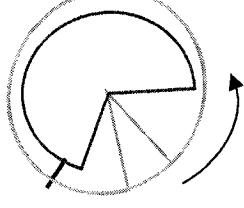
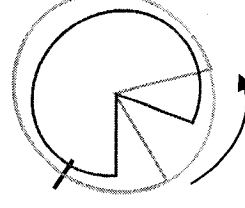
Again in this case to one can apply the swap invariance property to reduce the number of topologically unique combinations to 12. Table 5 shows the unique combinations. The 12 combinations for the C-axis along with the 3 combinations for the L-axis yield 36 combinations of overlaps for rectangles on surface of the cylinder. Figure 44 displays 9 such combinations based on which case is picked for the L-axis and C-axis respectively.



**Figure 43.** Interpreting an interval: Based on the placement of the start mark and direction of rotation, it is possible to encounter the end mark ( $e_1$ ) of an interval before the start mark ( $s_1$ ). The sequence of points maps to case 16 in Table 4 and the deduced intervals  $I_1$  and  $I_2$  are shown

Case No.	Ordering	Relationship	Case No.	Ordering	Relationship
1	$s_1-s_2-e_1-e_2$	 P	2	$s_1-s_2-e_2-e_1$	 Sp/Sb
3	$s_1-e_1-s_2-e_2$	 S	4	$s_1-e_1-e_2-s_2$	 Sp/Sb
5	$s_1-e_2-s_2-e_1$	 P	6	$s_1-e_2-e_1-s_2$	 P
7	$s_2-s_1-e_1-e_2$	 Sp/Sb	8	$s_2-s_1-e_2-e_1$	 P
9	$s_2-e_1-s_1-e_2$	 P	10	$s_2-e_1-e_2-s_1$	 P
11	$s_2-e_2-s_1-e_1$	 S	12	$s_2-e_2-e_1-s_1$	 Sp/Sb



Case No.	Ordering	Relationship	Case No.	Ordering	Relationship
13	$e_1-s_1-s_2-e_2$		14	$e_1-s_1-e_2-s_2$	
15	$e_1-s_2-s_1-e_2$		16	$e_1-s_2-e_2-s_1$	
17	$e_1-e_2-s_1-s_2$		18	$e_1-e_2-s_2-s_1$	
19	$e_2-s_1-s_2-e_1$		20	$e_2-s_1-e_1-s_2$	
21	$e_2-s_2-s_1-e_1$		22	$e_2-s_2-e_1-s_1$	
23	$e_2-e_1-s_1-s_2$		24	$e_2-e_1-s_2-s_1$	

**Table 4.** Relationships between two C-intervals based on the ordering of the points

Case No	Interval label		Case No	Interval label
1	$s_1-s_2-e_1-e_2$	P	13	$e_1-s_1-s_2-e_2$ Sp/Sb
2	$s_1-s_2-e_2-e_1$	Sp/Sb	14	$e_1-s_1-e_2-s_2$ P
3	$s_1-e_1-s_2-e_2$	S	15	$e_1-s_2-s_1-e_2$ P
4	$s_1-e_1-e_2-s_2$	Sp/Sb	16	$e_1-s_2-e_2-s_1$ S
5	$s_1-e_2-s_2-e_1$	P	17	$e_1-e_2-s_1-s_2$ P
6	$s_1-e_2-e_1-s_2$	P	18	$e_1-e_2-s_2-s_1$ P

**Table 5.** The 12 topologically unique cases of overlaps along the C-axis

The following definitions set up the framework to explain the concept of degree of partial overlap.

**Definition 1:** Any two rectangles that overlap with each other are *neighbors* of each other.

**Definition 2:** Neighbors that overlap each other partially are *partial neighbors*.

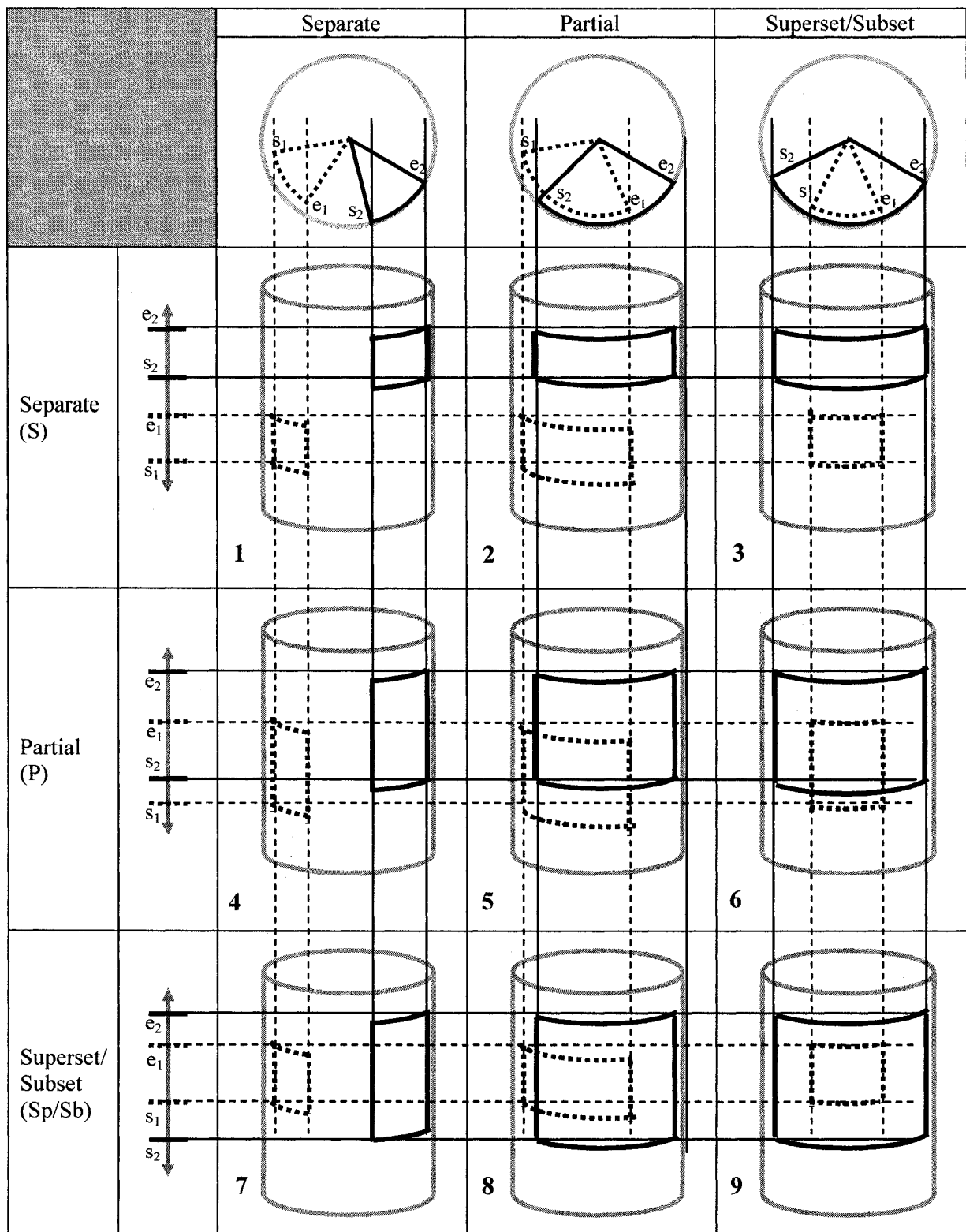
Case 9 in Figure 44 is an example of two rectangles that are neighbors of each other but not partial neighbors, whereas cases 5, 6, and 8 show examples of partial neighbors.

**Definition 3:** The *degree of partial overlap* of a rectangle is the largest number of partial neighbors it has, such that all these neighbors belong to the same agent.

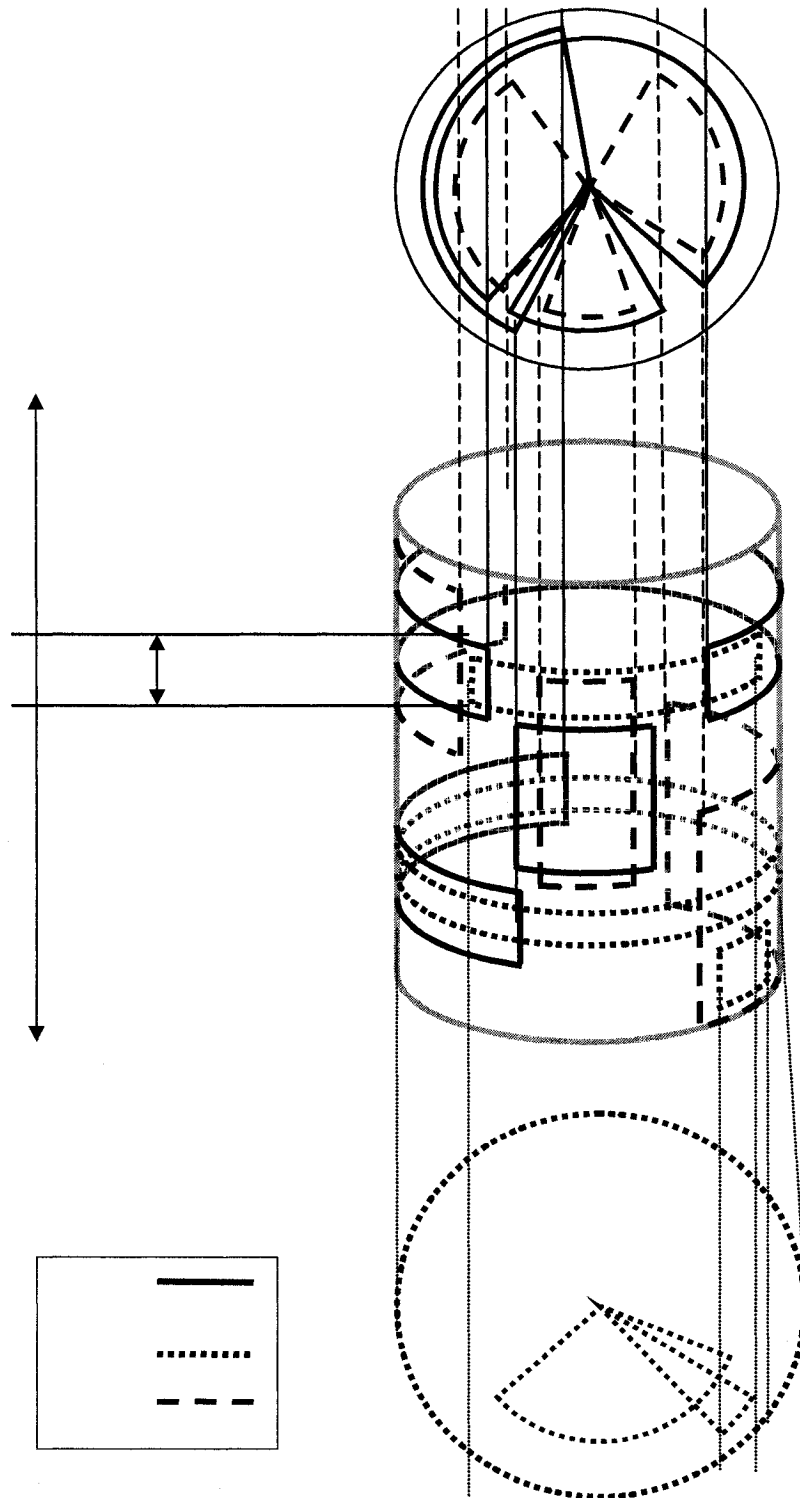
By verifying that the degree of partial overlap is not more than one, it can be ensured that not more than one rectangle of each agent overlaps partially with the rectangle under consideration. If this condition is fulfilled, then there exists a procedure that is guaranteed to allocate a rectangle to each agent such that the rectangle was marked by the agent itself. This procedure was explained in section 5.2.7 to solve the allocation of a two-dimensional planar resource. I illustrate the procedure through an example.

#### **6.3.4. An Example Allocation Problem**

I continue with the running example of three agents  $i, j$ , and  $k$ , which are contesting for a portion of the two-dimensional cylindrical resource. The agents have to mark out three rectangles, each, which are equal in their valuation as per the protocol. Based on the topology of rectangle overlaps (Figure 45), one computes that the degree of partial overlap is at most one, i.e., the overlap degree condition is fulfilled. Therefore the procedure will be able to find a feasible allocation.



**Figure 44.** The different types of overlap between two rectangles on a cylindrical surface. Only 9 of the 36 possible combinations are shown.



**Figure 45.** Agents  $j$  and  $k$  mark out portions on a cylindrical resource that fulfill the overlap degree condition. All projections along the C-axis are shown. Only one projection onto the L-axis is shown to reduce clutter.

The procedure given the overlap degree condition will be executed as follows:

1. The rectangles are submitted to a mediator that collects them in a list.
2. The L-co-ordinates of the intervals are read into the L-list. The procedure sorts the L-co-ordinates of the intervals in the order of their occurrence from bottom to top. The C-co-ordinates are sorted by (arbitrarily) picking one C-co-ordinate as the first one. Then moving in the counterclockwise direction one reads the points that are encountered into the C-list. The procedure will determine which rectangle needs to be allotted to an agent based on the orderings of the intervals. The actual position of the points on the cylindrical surface is not needed by the allocation procedure. Since there are no cardinal comparisons between agent rectangles (i.e., "areas" of rectangles are not computed), one does not require the resource to be measurable [47].
3. The relation of the L-intervals to each other is determined. The relations can be one of S, P or Sp/Sb. Thus each rectangle has a set of L-relations with its neighbors after one parses through the L-list. The procedure similarly processes C-list creating a set of C relations for each rectangle.
4. A scoring matrix is utilized, as shown in Figure 46. Note that the matrix has 16 values although 36 scenarios were mentioned earlier. This is because, in order to determine the relationship between any two rectangles, one needs only the relations they have with each other along the L-axis and C-axis respectively. And along each axis there exist only four possible values for the relations: S, P, Sp, Sb. Note that Sp(superset) and Sb(subset) are the same topology-wise. For example, if interval A is a superset (Sp) of interval B, then it is the same as saying interval B is the subset (Sb) of interval A. But the scoring matrix is created from the "point of view" of a particular rectangle.

Thus rectangle A being a subset of rectangle B is distinct from rectangle B being a superset of rectangle C. Hence Sp and Sb are treated as separate cases. Also note that the sufficient condition for any two rectangles not to overlap is that their respective intervals do not overlap in at least one of the axes. Since such rectangles are not neighbors of each other they are not assigned a score.

	<b>S</b>	<b>P</b>	<b>Sp</b>	<b>Sb</b>
<b>S</b>	--	--	--	--
<b>P</b>	--	<b>0</b>	<b>0</b>	<b>0</b>
<b>Sp</b>	--	<b>0</b>	<b>-1</b>	<b>0</b>
<b>Sb</b>	--	<b>0</b>	<b>0</b>	<b>1</b>

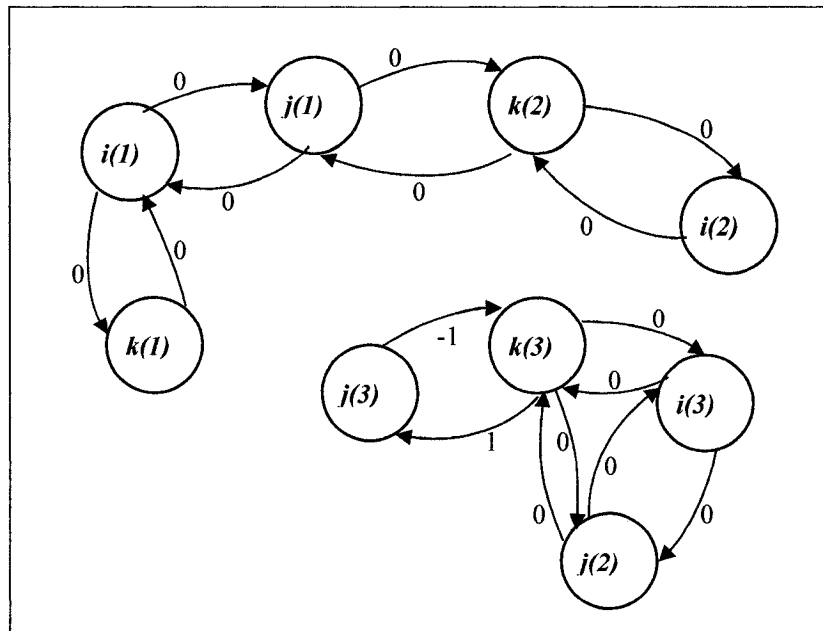
**Figure 46.** The scoring matrix for a rectangle. The types of interval overlaps are Separate(S), Partial(P), Superset(Sp), Subset(Sb)

5. Based on the types of overlap each rectangle has with its neighbors, one creates a directed graph that represents the connections. The directed graph is created as follows:

For any two rectangles,

- Each rectangle is represented as a node.
- If there is no overlap between the two rectangles, there is no edge between the corresponding nodes.
- If there is a partial overlap between the two rectangles, then each node will have a directed edge connecting the other node with an edge weight of 0.

- d. If rectangle A is a subset of rectangle B, the edge going from Node B to Node A will have a weight of 1, whereas the edge going from Node A to Node B will have a weight of -1. Refer Figure 47 for the graph corresponding to the example in Figure 45.



**Figure 47.** The directed graph based on the topology of overlapping rectangles

6. Once one has the topology of the connections in graph form, the procedure will arbitrarily start at some rectangle (i.e., node) and allot the particular rectangle based on the following conditions:
  - a. If there is no outgoing edge with edge weight of 1, then allocate this particular node.



- b. If there is such an edge, then travel along this edge and move to the node connected to it. Now let this be the new node under consideration. Repeat the procedure using this node as the starting point.

Once a particular node (i.e., the rectangle corresponding to it) has been allocated then remove the following nodes from the graph, for future consideration:

- c. The node just allocated.
- d. All neighbors of the current node.
- e. All the nodes belonging to the agent which is the owner of the current (allocated) node.

Repeat Step 6 till all agents have been allocated nodes (rectangles).

Applying step 6 to our example, choose  $i(1)$  (arbitrarily) to be the first node.  $i(1)$  has no outgoing edge with edge weight 1, hence it is allocated (step 6a). Next remove the following nodes:

- $i(1)$  (Step 6c)
- $k(1)$  and  $j(1)$  (Step 6d)
- $i(2)$ ,  $i(3)$  (Step 6e)

The following nodes remain:  $j(2), j(3), k(2), k(3)$  and agents  $j$  and  $k$  still have to be allocated their nodes (rectangles). So repeat Step 6 and arbitrarily choose  $k(3)$  as our starting node. Now  $k(3)$  has an outgoing edge with edge weight 1, so travel along the edge and move to node  $j(3)$ .  $j(3)$  has no outgoing edges with weight 1, so allocate this node. Next applying Steps 6c, 6d, and 6e remove the following nodes from consideration:

$j(3)$ ,  $k(3)$ ,  $j(2)$ . Only node  $k(3)$  is left which can be allocated to agent  $k$ . Thus each agent receives a fair share of the resource.

The proof that the procedure is guaranteed to come up with a fair allocation is described in section 5.2.7. Only the statement of the theorem is mentioned here.

**Theorem 3.** If there are  $n$  agents, and each agent makes  $n$  rectangles, creating  $n$  portions of a rectangular cake and if the rectangles are so marked that the degree of partial overlap is not greater than 1, then the procedure guarantees that each agent will be allotted a piece, such that the piece was one of the  $n$  portions created by the agent itself.

The following pseudo code gives an idea about the structure of the procedure:

```
while (graph.hasMoreNodes ())
    currentNode=graph.getNewNode ()
    while (currentNode.hasMoreEdges ())
        currentEdge=currentNode.getNewEdge ()
        if (currentEdge.getEdgeWeight ()==1)
            currentNode=currentEdge.getOtherNode (currentNode)
        end if
    end while hasMoreEdges
    currentNode.setAllocated (true)
    graph.remove (currentNode)
    graph .remove (currentNode.getNeighbors ())
```

```
graph.remove(currentNode.getOwner.getOwnedRectangles())  
end while hasMoreNodes
```

### ***6.3.5. Features of the Native Cylindrical RA Procedure***

No two nodes belonging to the same agent will ever share a common edge in the graph. This is because the agents have to adhere to the protocol and ensure that no two rectangles belonging to the same agent overlap. The procedure will be executed by the agent which volunteers to be the mediator. The agent will keep a record of the list of nodes it has visited. This allows for any other agent to run the procedure and confirm that the list of nodes traveled is valid. This is how one can verify that the allocation is fair and guard against mediator bias. Note that the set of allocated nodes depends on which node is chosen as the starting node. But it does not affect the feasibility of the allocation. Even though one chooses the starting node arbitrarily, it will always be guaranteed a feasible allocation because it was confirmed that the degree of partial overlap was at most 1.

I conclude this section with some brief discussion on the features of this procedure. This procedure is similar to the hill-climbing algorithm [93] that tries to find the "highest point" among various nodes. The procedure is also less restrictive in terms of how agents mark their rectangles. Once the mediator has verified that the degree of overlap is at most 1, a solution can be guaranteed. In case this condition is false, one can still run the procedure, but there is no guarantee that a feasible solution exists. The degree of partial overlap property is only a sufficient condition for the existence of a solution, it is not a

necessary condition. A deeper discussion of various issues involved with this procedure is presented in chapter 7.

#### **6.4. Conclusion**

This chapter presents protocols and procedures for allocation of two-dimensional cylindrical resources. These are resources that have the topology of the cylindrical surface. Such resources are encountered when one of their dimensions is finite and bounded (the axial dimension) while the other is finite and unbounded (the circular dimension). Some real world examples were presented to show the practical nature of the problem. The kind of cylindrical surfaces that were amenable to this protocol and procedure was discussed. Existing literature does yet explicitly cover such cases. It was shown how a cylindrical resource allocation problem could be mapped into the one-dimensional circular resource allocation problem and the appropriate procedure from chapter 4 could be applied. Next, a more flexible protocol was proposed wherein agents only marked  $n$  non-overlapping rectangles of equal value (by their valuation) on the cylindrical surface of the resource. If the non-overlap condition was true, then one could transform the existing allocation problem to a one-dimensional linear resource allocation problem which has already been solved. If the overlap degree condition is true I present a novel two-dimensional cylindrical resource allocation procedure which is guaranteed to come up with a solution. It uses the notion of degree of partial overlap to create a sufficiency condition for the existence of a solution, and proposes a procedure to come up with one in such a case. The proposed solution is fair, strategy-proof, constructive and

does not need the resource to be measurable. There exists considerable scope for future work, both in terms of improving the performance of the proposed procedure as well extending this work to apply to other topologies and domains.

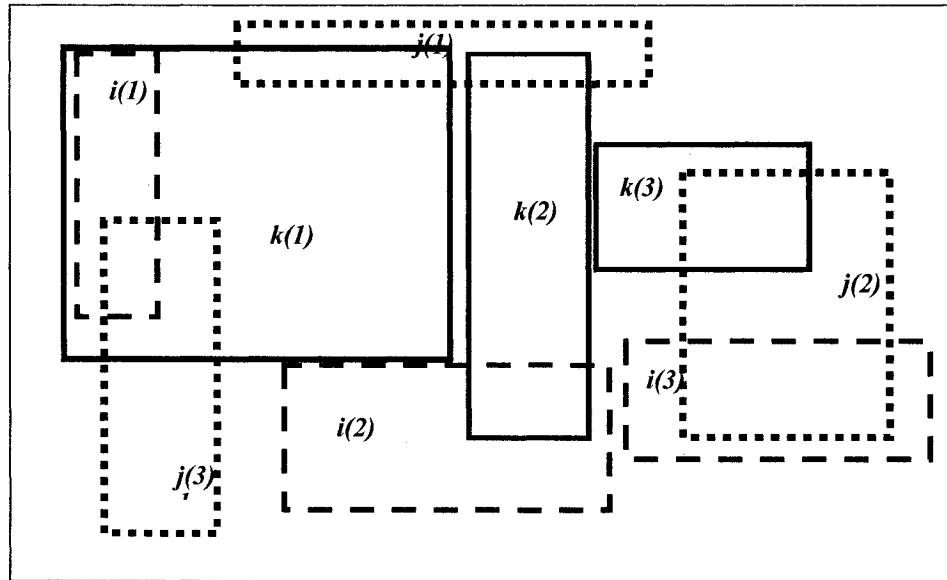
# Chapter 7

## Issues in Two-dimensional Resource Allocation Procedures

In this chapter, I discuss in greater detail various aspects of two-dimensional resource allocation procedure. The chapter is composed of two parts. The first part discusses how the solution spaces for the non-overlap condition and the overlap degree condition relate to each other. The procedure is analyzed with respect to various criteria like fairness, envy-freeness etc. in order to understand its benefits and drawbacks. The second part discusses future work relating to the two-dimensional resource allocation procedure. There exists considerable scope for extending and improving this procedure to apply to resources with other topologies and dimensions.

### 7.1. Discussion

What does the solution space look like with respect to the non-overlap condition and the overlap degree condition? In general the conditions can be true or false independent of each other. The examples below show all the possible scenarios that can occur and give a picture of how the sets are related to each other. I continue with the running example of chapter 5 and denote the agents as  $i, j$  and  $k$ .



**Figure 48.** The non-overlap condition and the overlap degree condition are true.

1. The non-overlap condition and the overlap degree condition are true (Figure 48)
  - The non-overlap condition: The X coordinates of the rectangles of each user do not overlap each other.
  - The overlap degree condition: The degree of partial overlap of each rectangle is at most 1
2. The non-overlap condition is true, but the overlap degree condition is false (Figure 49)
  - The non-overlap condition: The Y coordinates of the rectangles of each user do not overlap each other.
  - The overlap degree condition: The degree of partial overlap of  $i(1)$  is 3 (more than 1)

3. The non-overlap condition is false, but the overlap degree condition is true (Figure 50)

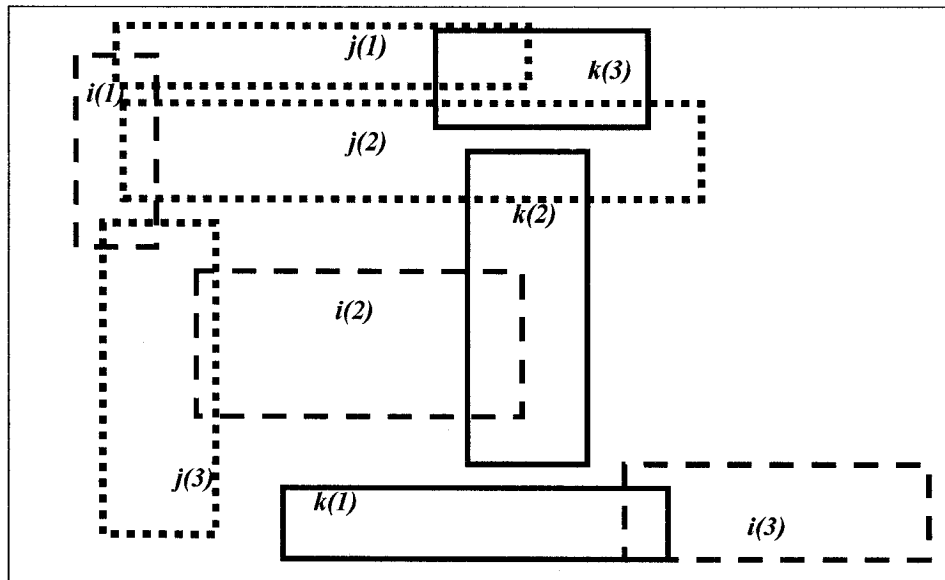
- The non-overlap condition: For agent  $j$ ,  $j(1)$  and  $j(2)$  overlap on the X-axis, but  $j(2)$  and  $j(3)$  overlap on the Y-axis.
- The overlap degree condition: The degree of partial overlap of each rectangle is at most 1

From the examples shown, one can conclude that the solution spaces for Iyer's conditions:

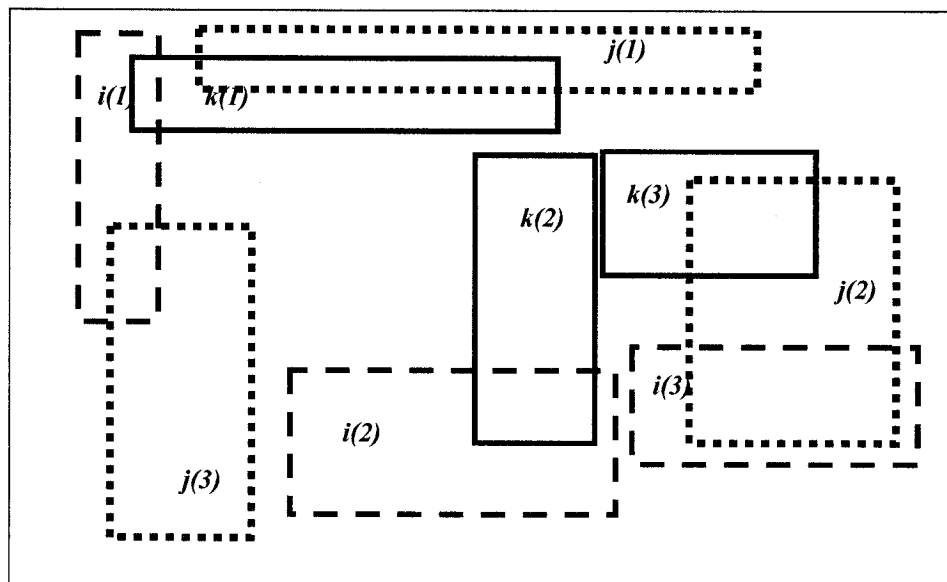
- are not subsets of each other.
- are not mutually exclusive.

Thus while the overlap degree condition can be taken as more flexible, it is not necessarily a weaker condition. One can look upon the overlap degree condition as a way to expand the space of solutions available to us. The solution space looks as shown in Figure 51.

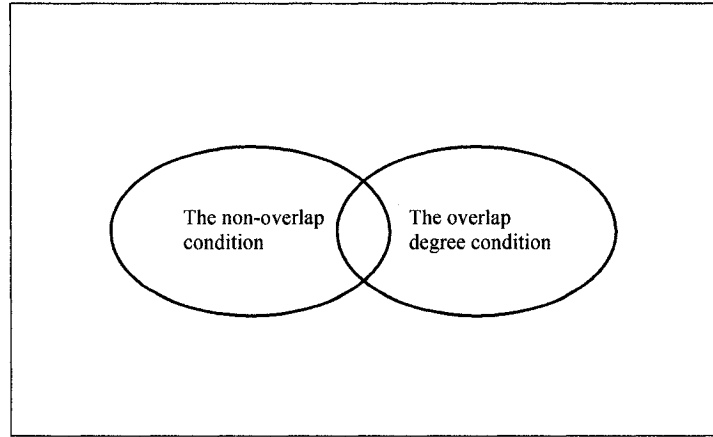




**Figure 49.** The non-overlap condition is true and the overlap degree condition is false.

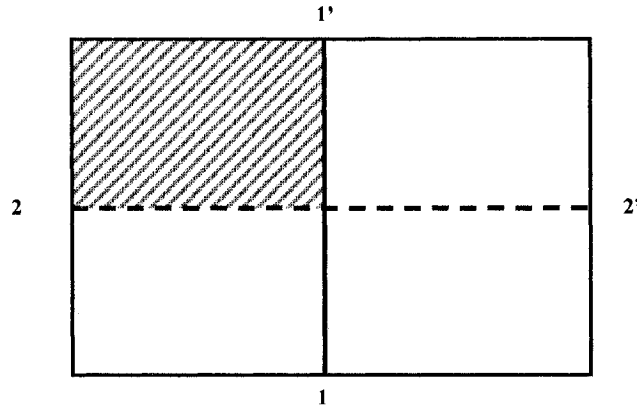


**Figure 50.** The non-overlap condition is false and the overlap degree condition is true.



**Figure 51.** The solution space for Iyer's conditions.

Either of these conditions tests for the presence of a solution, but their absence cannot be taken as a guarantee that a solution does not exist. Is it possible for any algorithm to guarantee an allocation if both of Iyer's conditions are false, even though the agents follow the protocol (in section 5.2.1 or section 6.3.2)? I show that there exists no algorithm that can guarantee a solution, if just the protocol is followed. Specifically I focus on step 3 of the protocol, which requires that the agents own portions do not overlap.



**Figure 52.** Agent 1 divides the land into two vertical strips, while agent 2 divides them into two horizontal strips.

**Theorem 4.** If all agents follow only the protocol, then there exists no algorithm that can guarantee an allotment of rectangles, such that each agent will get a rectangle marked by itself.

**Proof:** I prove the above statement by showing a contradictory example. Consider two agents who want a plot of land divided among them. The agents create rectangles as shown in Figure 52. In such a case no matter what interval is allocated to agent 1, it will conflict with an interval of agent 2 and vice-versa. □

What the theorem tries to formalize is the idea that in addition to the protocol, some condition is required that will guarantee a feasible allocation exists. Two such conditions have been put forth in Chapters 5 and 6. It is quite possible that one can come up with other conditions that may further extend the solution space. The ideal case would be if one could find the perfect condition X, which if fulfilled will include all the feasible allocations, i.e., it would be the largest superset of Iyer's conditions that covers the entire

solution space. This is quite unlike the one-dimensional case, where just following the protocol guaranteed that the solution could be found. One discovers that if the resource is two-dimensional it is impossible to guarantee a solution for every possible permutation of agent rectangles. Ensuring that the agents follow the protocol alone is not a sufficient condition for a solution to exist. As future work, one can look into the possibility of discovering conditions that may be able to exhaustively cover the solution space.

In the two person case shown above, there exists a procedure that tends to fairness, if one allows multiple iterations of the procedure to run. In the example shown, suppose the allocation procedure allocates the left column to agent 1 and the top row to agent 2 based on their markings, then one gets the shaded region as the disputed area. Now the agents will again be required to repeat the procedure of markings, but this time only with the smaller disputed area. After a finite number of iterations the procedure stops, once agents can feel satisfied that they have a piece, at least as large as the other within the limit of tolerance. If the resource being divided is not recombinable (like land), successive applications of the procedure may create allotted portions that are not contiguous with previously allotted portions. Thus there may be isolated portions of land belonging to agent 1 surrounded by land belonging to agent 2. Hence the iterative solution may only be applicable in qualified situations.

Next, I discuss how well the procedure compares with various criteria mentioned in the literature survey. Specific references are cited for each criterion in case the reader needs further clarification of the concepts.

- **Fairness** [14]: The proposed protocol is fair. As mentioned earlier, an agent will find the allocation *fair*, if it feels it got exactly  $1/n$  of the value of the entire resource. Since as per the protocol the agent divides the resource into  $n$  rectangles of equal value and the procedure (given the non-overlap condition and/or the overlap degree condition is true) will allot one such rectangle to that agent, the agent gets  $1/n^{\text{th}}$  (by its own valuation) of the entire resource. Thus the two-dimensional procedure allocates resources in a fair manner.
- **Envy-freeness** [14]: Envy-freeness means that every agent thinks that the portion allocated to it is greater in value than the portions other agents received. Thus no agent envies another agent's portion. It is strongly desirable for procedures to be envy-free but it is tougher criteria to fulfill than just fairness. There exist very few procedures that can guarantee envy-free division. The two-dimensional procedure while fair is not envy-free. It is quite possible that though an agent may consider the value of the portion it received to be  $1/n$  of the total (and hence fair), it may be envious of another agent whose portion it thinks is of greater value than its own.
- **Efficiency**: Efficiency is also a criterion for judging procedures. It provides a measure of how much of the resource is wasted in the process of allocation. The term "efficient" is used in the pareto-optimal sense, i.e., is it possible that there exists an alternative allocation in which at least one agent is better off while the others are no worse than the current allocation? The answer in the case of the two-dimensional procedure is yes, i.e., the procedure is inefficient at allocating resources. This is because all the portions allocated to agents do not exhaustively use up the resource. The "leftovers" can be redistributed among agents by running the protocol again and

letting interested agents submit their preferences. This can be done repeatedly until finally every agent thinks the remnants are negligible in value. Such an iterative version of the protocol will certainly help improve the efficiency of resource distribution. However the effectiveness of such an approach depends on the type of resource being distributed. If the resource is *infinitely divisible* and *recombinable* (like a cake) then such an iterative procedure is appealing. But some resources like scheduling time on a supercomputer users or land division may only be infinitely divisible (though not recombinable) and in such cases the repeated divisions of the leftovers will make the agents portions so small as to be basically worthless. The two-dimensional protocol and procedure is however more efficient than the traditional one-dimensional cake-cutting, especially when agents value some portions of the resource negatively. This is because the agents are allowed to cut in two dimensions, thus completely eliminating negative utility regions from being included in the allocation. While there is a clear need to improve the efficiency of the single shot protocol and procedure, it needs to be matched up against the concurrent increase in the complexity of the protocol as well as the procedure.

- **Complexity:** What is the space and time complexity of this procedure? Let us calculate the space complexity first. Each rectangle is fixed by four points, with each point having two values (one each for the C-coordinates and L-coordinates respectively). Thus each rectangle is described by eight values. Each agent marks  $n$  rectangles, creating a total of  $n^2$  rectangles (by  $n$  agents). Thus the space complexity of the procedure is  $8n^2$ . Now I compute the time complexity. First the list of C-coordinates and L-coordinates needs to be sorted. An efficient sorting algorithm,

such as insertion sort, will take  $n \log n$  time, which translates to  $2n^2 \log n$  time (replacing  $n$  by  $n^2$ ) for each axis. Next C and L relations need to be created. Since the overlap degree condition is true, each rectangle can have at most  $n-1$  partial neighbors. If one rectangle is completely subsumes the other (i.e., they are neighbors but not partial neighbors), then the inner rectangle can have at most  $n-2$  partial neighbors. Thus the rectangles will have an average of  $n-1$  neighbors in the worst case. Thus a total of  $n(n-1) = n^2 - n$  relations are possible on each axis. Analogously the graph created out of these relations will have vertices equal to  $n^2$  and number of edges equal to  $n^2 - n$  (in the worst case). The running time of the procedure to traverse these nodes is therefore  $O(n^2)$ .

- **Strategy-proof [7]:** A protocol is said to be *strategy-proof*, if for each agent declaring its true evaluation is a dominant strategy. A typical example of a strategy-proof mechanism is the Vickery auction where agents submit sealed bids to the auctioneer and the agent quoting the highest price wins but is only required to pay the amount of the second highest bid. The planar and cylindrical resource allocation protocol is strategy-proof. The dominant strategy for any agent while drawing out rectangles is to make sure that it values each of the rectangles it draws as exactly  $1/n$ . If an agent tries to draw a rectangle, that is smaller in value than any other agents' rectangle, in order to guarantee itself a specific piece of the resource, it may end up getting the piece, but the value of that piece will be smaller than  $1/n$ , thus getting less than its fair share. On the other hand, if an agent tries to get more than  $1/n$  by drawing a rectangle as large as possible, then it is quite likely that it will subset some other agents' rectangle which will end up getting allocated to another agent. For

clarification consider the following extreme example: Agent  $i$  is greedy and wants as much of the resource as possible. So it draws one large rectangle which covers the entire resource, while other agents follow the dominant strategy and draw  $n$  rectangles of equal value. What will agent  $i$  receive? Agent  $i$  receives nothing because the procedure that allots rectangles takes as input one large rectangle (which agent  $i$  had drawn) and  $n-1$  rectangles of zero value and the procedure ends up allocating one as these zero valued rectangles as agent  $i$ 's allocation. Thus agent  $i$  gets zero utility by being greedy. The protocols are therefore not open to manipulation by greedy agents and attempts to misreport preferences will backfire. There is no incentive for the agents to lie and hence the protocol is strategy-proof.

- **Measurability** [47]: This is an important notion for mathematicians and economists. The basic assumption is that any resource being allocated must be “measurable,” i.e., there must exist a function that can assign a number (like “length,” “area,” or “volume”) to subsets of a given set (like land for example). This notion paves the way for a cardinal comparison of various subsets for enabling an allocation. However the two-dimensional procedure does not need cardinal comparisons to make an allocation. I create portions based on the topology of overlaps. If the one rectangle is completely contained in another then the smaller rectangle is allocated. If two rectangles only partially overlap each other then the one that was encountered first (by the procedure) is allocated first. There is no notion of comparison of agent rectangles based on their “areas.” While agents may need the set to be measurable in order to “measure” out equal portions of the resource, such a restriction is not imposed by the protocol itself. Agents may use alternate means to create equal



portions, like creating  $n$  rectangles arbitrarily and adjusting their sizes iteratively till it doesn't prefer any one over the other. This is a way to *ordinally* create  $n$  rectangles of equal value without using the notion of a "measure." Unlike earlier work in land division, It is not explicitly required that the resource to be allocated be measurable.

- **Constructive (Non-existential):** The procedure proposed here is algorithmic. Most of the early literature dealing with two-dimensional resources (primarily in the form of land division) is dominated by economists and mathematicians. Their results are primarily existential in nature viz. they analyze various features of the problem (like types of utility functions, fairness and efficiency of the possible allocations etc) and show whether or not allocations "exist" with given properties. They do not however propose ways to find such solutions. Thus it is yet unknown that even if feasible solutions exist whether the problem of finding an allocation procedure is tractable. Because of a lack of constructive solutions to allocation problems it is not possible to implement them in the real world where agents negotiate with one another and a mediator is able to compute a feasible allocation in a reasonable amount of time. For multiagent system designers, in addition to the knowledge that feasible allocations exist, the following points need to be considered:
  - (a) Is there a procedure to find these allocations? (Yes)
  - (b) What is the communication cost of the procedure? ( $O(n^2)$ )
  - (c) What is the computational complexity of the procedure? ( $O(n^2)$ )
  - (d) Can the agent preferences be kept private? (Yes)
  - (e) Can the procedure be manipulated by agents' lies? (No)
  - (f) What is the efficiency of the allocation procedure? (Inefficient)

(g) Are the procedure and the protocol iterative? (No)

(h) Are the agents restricted in using certain classes of utility functions? (No)

I answer some of the questions here. This is the first result that can be expressed as a computer algorithm for allocation of two-dimensional planar and cylindrical resources. Not only does it test for the existence of a solution, it is able to find one if it exists (given the non-overlap condition and/or the overlap degree condition is true).

- **Nature of agent utility functions:** The survey of earlier literature shows a number of restrictions placed on agent utility functions none of which are applicable in our case. The agents may draw rectangles based on their internal utility functions or completely do away with the use of utility functions. As stated in the discussion earlier about measurability, agents can also draw rectangles using ordinal rather than cardinal comparisons. Briefly, I mention the typical restrictions on agent utility functions historically mentioned in literature, which are *not* applicable to our case:

(a) Non-atomic: This requirement specifies that the agent utility functions smoothly go down to zero as the amount of the resource tends to zero, i.e., there should be no “atoms” in the resource, where portions smaller than the “atom” are of zero value to the agent.

(b) Additive: This is common requirement for utility functions and helps simplify a lot of analysis. If A and B are two disjoint subsets then simple additivity states that:

$$v(A \cup B) = v(A) + v(B) \text{ for all disjoint } A, B \in \Sigma$$

- (c) Concave: Concave domains are a subset of subadditive domains and the utility functions have the property:

$$v(A \cup B) \leq v(A) + v(B) \text{ for all disjoint } A, B \in \Sigma$$

They are less commonly modeled than simple additivity because it is difficult to prove results in this domain. However they more realistically reflect the utility of obtaining additional resources in the real world.

- (d) Continuous: Utility functions are assumed to be continuous. This means that for every amount  $x$  of the resource the agent has a unique value for that amount. This simplifies analysis but may not be necessarily true in the real world. In addition, utility functions are said to be smooth if there are no “kinks” in the function value anywhere i.e., the function is differentiable at all points of the curve.

## 7.2. Future Work

This work presents protocols and procedures for the allocation of a two-dimensional planar and cylindrical resources. Chapter 4 proposed an allocation scheme for one-dimensional linear RA and circular RA problem. There is however a lot of scope for future works with this work as a starting point. The protocols and procedures can be extended to apply to other domains, such as resources that are topologically different from a cylinder (a torus, or a sphere for example). Further research will also help improve

the procedure by reducing the running times and allowing more generalized shapes. The pertinent issues that need to be looked at are mentioned here.

- **Increasing the space of feasible allocations:** The planar and cylindrical protocols in their current form are single shot protocols. Either the agents rectangles are laid out in a manner where feasible allocations exist or no agent gets any part of the resource (a conflict deal is reached). There is a need for creating an iterative version of the protocol whereby agents can send proposals to one another by continuously changing the shape of their rectangles and finding a solution that is feasible (and preferably optimal) to all the agents. It is quite likely that there will be a concomitant increase in the complexity of the protocol, communications, and the procedure. Thus while the space of solutions will certainly increase, mechanism designers will have to balance this against increased cost of computation both for the agents as well as the mediator.
- **Search for condition X:** A more generic condition than the two that have been proposed will serve to increase the space of feasible allocations. Iyer's conditions serve as a benchmark against which future conditions can be compared. The conditions are *sufficient* but not *necessary* conditions for the guaranteeing the existence of a solution. There can exist other conditions which behave similarly. The ideal condition X would be one that is both sufficient and necessary. In such a case condition X would exhaustively cover the solution space and would be the (largest) superset of both solutions covered by Iyer's conditions.
- ***n*-dimensional resources:** In this work I considered the case where a resource is a generalized two-dimensional plane or cylinder. Results of one-dimensional resource allocation were extended to two dimensions. But there were significant qualitative

differences between the problems. I presented proof that if the overlap degree condition is true then a feasible allocation exists and the procedure would be able to find the allocation. One can fashion proofs in an analogous manner for three (and higher) dimensional resources. But the challenge is to prove that the proof for the overlap degree condition holds for arbitrary  $n$ -dimensional resources. Is it possible to create a (meta) proof to prove a feasible allocation exists — if the overlap degree condition is true — for an  $n$ -dimensional resource with arbitrary  $n$ ?

- **Efficiency:** The current protocols while being more efficient than one-dimensional resource allocation schemes (like moving knife) are inefficient because there will always be leftovers from the allocation. As mentioned earlier, efficiency can be improved by creating a simple iterative version of the single shot protocols. Any work done to improve the efficiency of allocation will greatly increase the appeal of these procedures but it is likely to increase the complexity of the protocol as well the procedure in the process.
- **Allowing more general shapes:** The protocols as they are presented currently allow agents to mark regions of interest in the shape of rectangles only. This no doubt simplifies the allocation procedure but it is quite restrictive. If agents were allowed to mark out regions of interest as more general polygons then it will improve the efficiency of the system and increase the utility received by agents. It may also increase the space of feasible allocations. This is a challenging problem. If a procedure were found which is able to allocate for general polygons then it might be possible to extrapolate those results to the case where agents can draw amoeba shaped regions and the procedure can find a satisfactory solution.

- **Allowing other topologies:** I assume the resource to be a two-dimensional planar or cylindrical surface. Would it be possible to extrapolate the procedure to other shapes like the surfaces of spheres or torii? The procedures are not directly applicable to such shapes because of their different topology. But one can easily visualize resources having such topologies; for example: dividing underwater mineral resources may require the agents to take into account that the earth has the topology of a sphere rather than approximate it as a flat plane.
- **Creating a decentralized version of the procedure:** The procedures in their current form are centralized. Since the running time is  $O(n^2)$ , it is quite appealing to create a distributed form of this procedure, so that running time of the procedure is reduced. While creating a distributed version of the procedure is a non-trivial task in itself, the following issues need to be considered as well:
  - a) Is a different set of agents needed to execute the distributed procedure? Or can the participating agents themselves execute the procedure?
  - b) How can the information about agent preferences be kept private?

The intention of this chapter is to present an analysis of the two-dimensional resource allocation procedures. While the non-overlap condition and the overlap degree condition are related to each other, they do not exhaustively cover the entire solution space. A discovery of a more general condition  $X$  that covers the entire solution space will be useful. The two-dimensional procedures are fair, strategy-proof, constructive and do not require the resource to be measurable. Scope for future work exists in the form of

improving the running times of the procedures as well as extending them to be applicable to other topologies and domains.

# Chapter 8

## On the Applicability of Sperner's Lemma for Multiagent Resource Allocation

Sperner's lemma is a simple but powerful combinatorial result that can be used to solve problems in multiagent resource allocation. This chapter discusses the applicability of Sperner's lemma in a multiagent system framework. I discuss the mechanics of how Sperner's lemma works, and then discuss an earlier result [20] that uses the lemma to attain an approximate envy-free solution. Next, an alternative way of applying Sperner's lemma to the multiagent resource allocation problem is put forth that has lower communication costs. This result is not approximate envy-free, but it is approximate-fair. I discuss the conditions under which such solutions exist. A tougher problem to crack has been to come up with a constructive algorithm that can find efficient allocations. Finally, I discuss the problems that need to be solved before Sperner's lemma can be fruitfully used in multiagent resource allocation problems.



## 8.1. The Mechanics of Sperner's Lemma

Sperner's lemma is a simple but powerful combinatorial result first stated by Sperner [94]. A good explanation of the intuition behind Sperner's lemma is presented by Francis Edward Su [20]. It is repeated here in a more concise fashion to create the context for later sections of this chapter. It can be stated as:

### Sperner's Lemma for Triangles

Any Sperner labeled triangulation of a triangle  $T$  must contain an odd number of elementary triangles possessing all labels. Specifically, at least one such elementary triangle must exist.

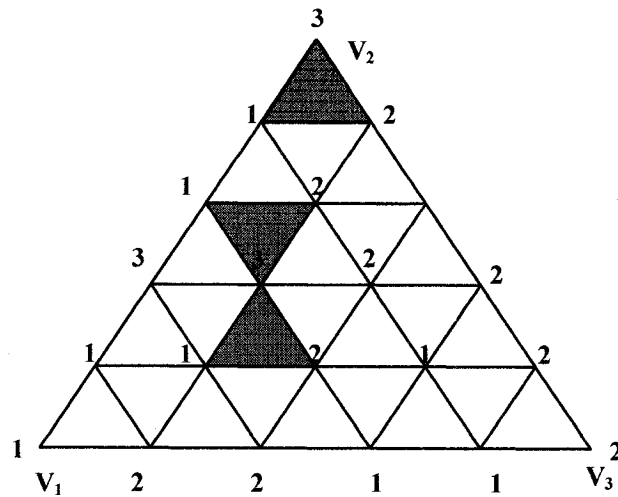
Refer to Figure 53. The standard triangle is called the 2-simplex. A 3-simplex would be a tetrahedron, whereas a 1-simplex would be a line. The "triangle" mentioned in the lemma can be generalized to any  $n$ -simplex. The  $n$ -simplex is defined as follows:

"An  $n$ -simplex is the convex hull of  $n+1$  affinely independent points."

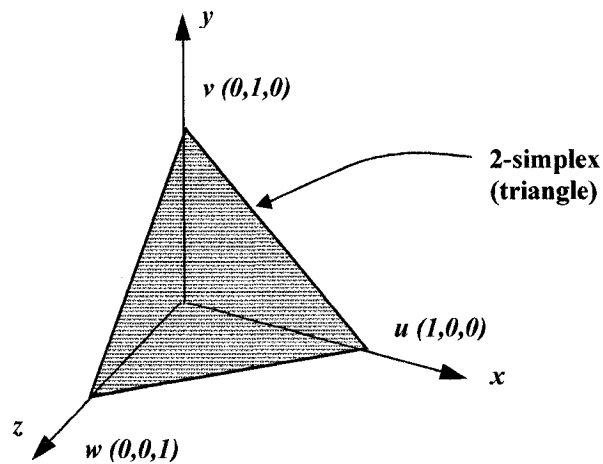
A simple example of "affinely independent points" are three points  $u$ ,  $v$ , and  $w$ , which have non-zero  $x$ ,  $y$ , and  $z$  coordinates. In vector notation:

$$u = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}; v = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}; w = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

The convex hull of  $u$ ,  $v$ , and  $w$  therefore forms a two-dimensional triangle (i.e., a 2-simplex) embedded in three-dimensional Euclidean space, as shown in Figure 54. Similarly, a 3-simplex would be a tetrahedron embedded in 4-dimensional space. Thus one can embed an  $n$ -simplex in  $n+1$ -dimensional space. I use the 2-simplex (triangle) as a running example to explain various concepts of Sperner's lemma.



**Figure 53.** Triangulation of a 2-simplex (triangle) into elementary simplices. The (1,2,3) represent Sperner labeling of the triangulation.



**Figure 54.** An example of a 2-simplex (triangle) embedded in 3-dimensional Euclidean space.

### 8.1.1. Triangulation

The interior of any  $n$ -simplex can be divided into smaller elementary  $n$ -simplices. The interior is to be covered exhaustively in this manner and elementary simplices should not intersect with each other except in a common face (if they are neighboring simplices). Such a set up is called the *triangulation* of the  $n$ -simplex. Figure 53 shows the triangulation of the 2-simplex into many smaller elementary simplices.

An  $n$ -simplex can be built up inductively as an assembly of  $n+1$   $n-1$ -simplices, which form its “facets.” Thus a 4-simplex (tetrahedron) can be assembled from four 3-simplices as its facets. Each 3-simplex (tetrahedron), in turn is built from three 2-simplices as its facets. Finally, each 2-simplex (triangle) is built from two 1-simplices (i.e., vertex points).

The power of Sperner's Lemma derives from the fact that one can inductively transmit properties known to be true in lower dimensions upward to higher dimensions. Similarly, the process of triangulating an  $n$ -simplex automatically triangulates each  $n-1$ -simplex that forms its facet.

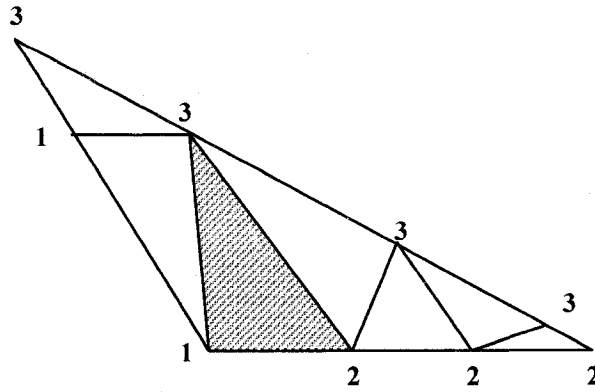
### 8.1.2. Sperner Labeling

Once the  $n$ -simplex has been triangulated, one can apply Sperner labeling to all the vertices as follows:

1. The vertices of the main  $n$ -simplex are uniquely labeled. Denote the main vertices as:  $V_i, i = 1, 2, \dots, n+1$
2. The interior of any  $k$ -simplex – formed as a convex hull of  $k+1$  main vertex points ( $V_k$ ) – can be labeled with any of the  $k+1$  labels carried by the vertices.

Any elementary  $n$ -simplex that carries all  $n+1$  labels on its vertices is called a *fully labeled elementary simplex*. Sperner's lemma states that there is at least one such simplex and, more generally, an odd number of such simplices exist. They are shown by the shaded elementary triangles in Figure 53. The Sperner's lemma for any  $n$ -simplex can therefore be stated as:

“Any Sperner-labelled triangulation of an  $n$ -simplex must contain an odd number of fully labeled elementary  $n$ -simplices. In particular, there is at least one.” [20]



**Figure 55.** An arbitrary triangulation of a triangle. The vertices are labeled as per Sperner labeling rules

A constructive proof for this has been elegantly explained by Su [20]. Finally, note that since Sperner's lemma is a topological result, the geometric shape of the triangles does not matter. Similarly, the elementary simplices that subdivide the main simplex can be triangles of arbitrary shape and size. Thus, any arbitrary triangulation will do. Figure 55 shows such an example, along with the associated Sperner labeling. The fully labeled elementary triangle is shaded.

Note that the Sperner labeling can be applied inductively; i.e., to Sperner-label an  $n$ -simplex, one needs to ensure that the  $n+1$   $(n-1)$ -simplices that form its facets are Sperner-labeled first. Thus first begin by applying Sperner-labeling to a  $0$ -simplex (the vertex point), then proceed to higher dimensions until one finally gets the Sperner-labeling for the  $n$ -simplex. Consider the triangle in Figure 53. The three main vertices are the  $0$ -simplices. Sperner labeling can be achieved trivially by labeling them 1, 2, and 3 respectively. Next, the lines 1-2, 2-3 and 1-2 (i.e., the  $1$ -simplices) will have their interior vertices labeled with one of labels belonging to their facets (i.e., the main vertices).

Hence every vertex in the interior of line 1-2 will be labeled as a 1 or 2, but not 3. Vertices in the interior of lines 2-3 and 1-3 will be labeled similarly. Finally, the interior vertices of the triangle 1-2-3 can carry any one of the three labels.

### 8.1.3. Procedure for Locating a Fully Labeled Elementary Simplex

At least one fully labeled elementary simplex needs to be located. It is explained later in this chapter how such an elementary simplex provides an approximate-fair solution to the problem of resource allocation in a multiagent system. The procedure is outlined through the example shown in Figure 53. Formal descriptions of the procedure are provided by Kuhn [95] and Cohen [96]. Note also that Sperner's lemma also applies inductively. Thus in case of the example in Figure 53, the triangle (2-simplex) will have an odd number of full labeled elementary triangles. Similarly, the lines (1-simplex) bounding the triangle, will have an odd number of fully labeled elementary line segments. Specifically, note that there are three line segments with the label 1-2 for line 1-2. Finally, for the case of the three main vertex points (0-simplex), Sperner's lemma is trivially true.

The procedure for locating a fully labeled elementary triangle then can be applied as follows:

**get** triangulated  $n$ -simplex  $T^n$

**for** each fully labeled elementary 1-simplex  $\tau^1$  in  $T^1$

$(T^1$  is the 1-simplex labeled 1-2)

```

set  $k$  to 2  $(k$  denotes the dimension number)

while  $k < n$ 

    call compute- $k$ -simplex with  $\tau^{k-1}$  returning  $\tau^k$ 

     $(\tau^{k-1}$  will have labels  $1-2-\dots-k$ )

    if  $\tau^k$  is fully labeled  $(i.e., \tau^k$  is labeled  $1-2-\dots-k-k+1$ )

        set  $\tau^{k-1}$  to  $\tau^k$ 

        increment  $k$ 

    else

        set  $\tau^{k-1}$  to the other facet of  $\tau^k$  with label

         $1-2-\dots-k$ 

    endif

endwhile

endfor

```

The procedure “*compute- $k$ -simplex*” has the following pseudocode:

```

get  $k-1$ -simplex  $\tau^{k-1}$ 

if  $\tau^{k-1}$  lies on  $I^{k-1}$   $(I^{k-1}$  is a facet of  $I^n$  and lies on its “surface”)

    set  $\tau^k$  to null

endif

set  $\tau^k$  to the elementary  $k$ -simplex which has  $\tau^{k-1}$  as its

facet

set newVertex to the vertex in  $\tau^k$  but not in  $\tau^{k-1}$ 

if label of newVertex is not one of  $(1, 2, \dots, k)$ 

    set  $\tau^k$  to null

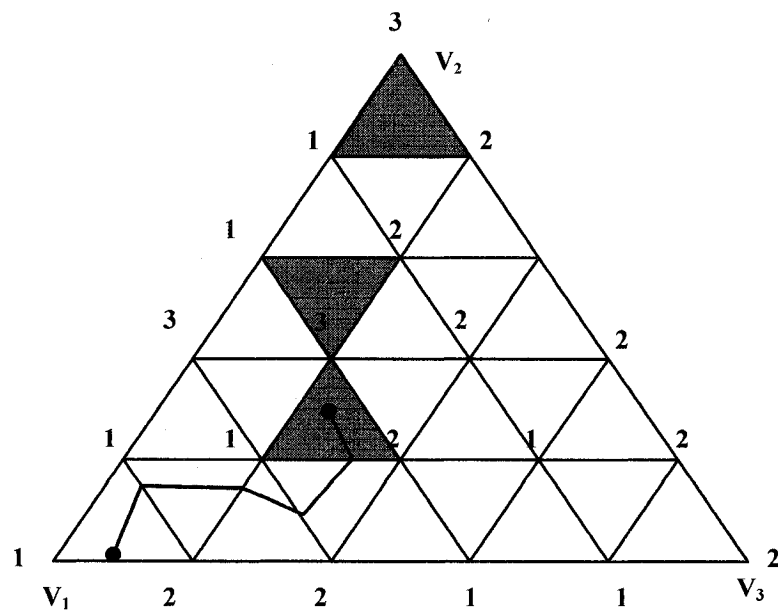
```

```

endif
return  $\tau^k$ 

```

Figure 56 visualizes the procedure. Note that not every fully labeled elementary simplex is accessible through this procedure. In section 8.3 I discuss in greater detail the various issues associated with the simplices found by this procedure.



**Figure 56.** Kuhn's procedure to locate a fully labeled elementary simplex.

## 8.2. Applying Sperner's Lemma to Multiagent Resource Allocation

Su [20] presented a scheme to exploit the properties of Sperner's lemma to enable resource allocation among  $n$  players. This result is immediately applicable to multiagent



resource allocation problems by changing some of the terminology. Although Su's solution was proposed for humans competing for scarce resources, one can easily modify the terms and apply it towards problems in multiagent systems.

### 8.2.1. Conventions and Assumptions

Before beginning a formal description of cake-cutting procedures using Sperner's lemma, I describe some conventions and assumptions. A knife - held parallel to one pair of edges of a rectangular cake - is moved slowly over it from the left edge of the cake to the right edge. The total size of the cake is 1 and the *absolute measure* of the  $i$ -th piece is  $x_i$ . By absolute measure I mean some metric like the "size," "area," or "length" of the cake that is universally agreed upon by agents as a way to determine the quantity of the resource. Thus:  $x_1 + x_2 + \dots + x_n = 1$  and each  $x_i \geq 0$ . Su's proposal therefore requires the resource to be Lebesgue measurable [47]. Other assumptions are that the utility functions of the players be non-atomic, positive, and additive. Atomicity deals with the aspect that however small a resource may be sliced, it must have some positive value for every agent, i.e., there should be no "atoms" in the resource, where portions smaller than the "atom" are of zero value to the agent. I also assume that the utility function is positive at all points along the cake. This ensures that players prefer any finite-sized piece to an empty piece of the cake. Additivity for utility functions can be stated as:

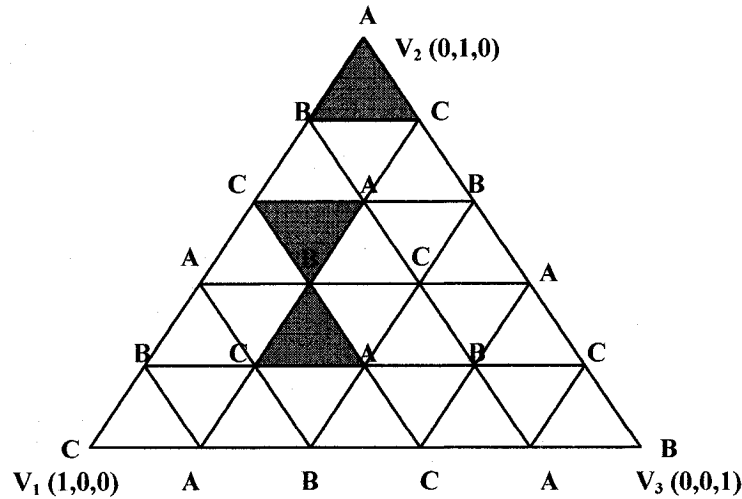
$$v(A \cup B) = v(A) + v(B)$$

i.e., incrementally adding to a portion should increase its value by a proportional amount.

### 8.2.2. *Approximate Envy-free Procedure for Multiagent Resource Allocation*

The following procedure is due to Su [20]. I now show how, given the assumptions and conventions about the agents, one can map the agent preferences into a correct Sperner labeling of the triangulation of an  $n-1$ -simplex. Once that is obtained, it will be shown how a fully labeled simplex represents an approximate envy-free allocation of the resource to various agents. An allocation is *approximate envy-free* if no agent thinks its portion is **smaller by  $\epsilon$  than** the largest portion in the allocation.

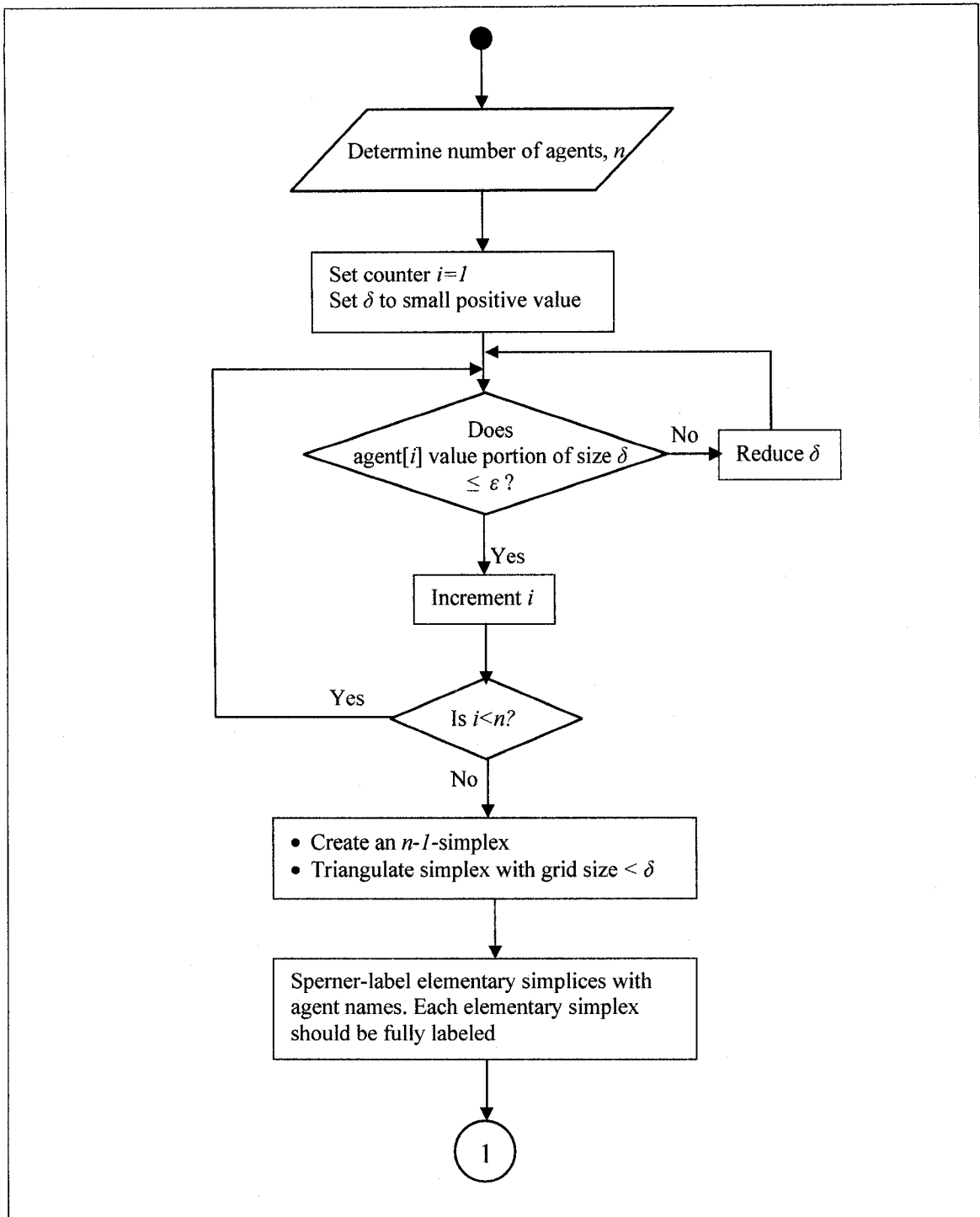
The space of possible allocations with the constraint  $x_1 + x_1 + \dots + x_n = 1$  forms a standard  $n-1$ -simplex in  $n$ -dimensional Euclidean space as shown in Figure 54. Each vertex's "ownership" is assigned to each of the  $n$  agents by using agent names as labels. Next, triangulate the simplex into many smaller elementary  $n-1$ -simplices. Label the interior vertices of the  $n-1$ -simplex by agent names and follow the rules of Sperner labeling. Although the labels in the interior of the  $n-1$ -simplex can be any of the ones on the main vertices of the simplex, the possibilities are further constrained in such a manner that every elementary simplex is completely labeled. For example, consider the case when  $n=3$  agents. They can be represented as the three vertices of the triangle (2-simplex). Figure 57 shows the necessary labeling of the vertices with A, B, and C being the agent names. Note that every elementary simplex carries all three labels.



**Figure 57.** Labeling of the triangulated 2-simplex with agent names. A, B and C denote the names of the agents

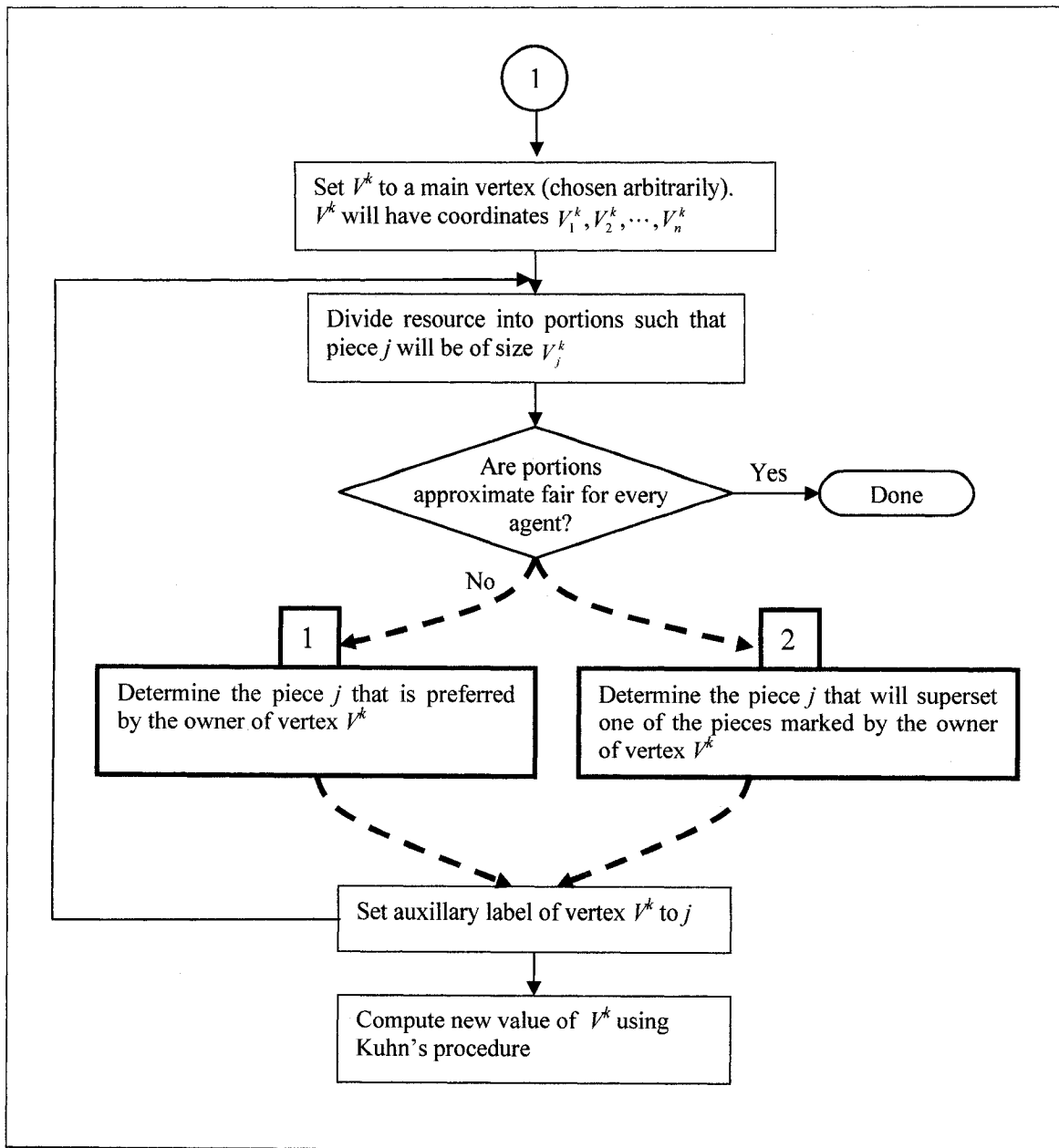
One can then achieve the auxiliary labeling of the elementary  $n-1$ -simplices in the following manner: ask the owner of each vertex which piece it would prefer out of the set of pieces of the resource corresponding to the location of the vertex. Thus for some vertex  $v^k$ , its co-ordinates will be  $(v_1^k, v_2^k, \dots, v_n^k)$  and piece  $j$  will be of size  $x_j = v_j^k$ . If the owner of the vertex, picks  $j$  as its preferred piece (because it views it as the largest) then that vertex is labeled  $j$ . Note that the auxiliary labels determined this way obeys the rules of Sperner labeling. Thus, each of the main vertices will be uniquely labeled by one of the  $j=1, \dots, n$  labels. The facets of the  $n-1$ -simplex, would carry one of the labels of the  $k$  ( $1 \leq k \leq n$ ) main vertices spanned by that facet. Figure 53 shows the example of such a Sperner labeling for  $n=3$  agents. By Sperner's lemma one is guaranteed that there exists at least one fully labeled elementary simplex in this triangulation. Each such elementary simplex represents an approximate envy-free allocation of portions to agents. In order to ensure that no agent thinks its portion is smaller by  $\epsilon$  than the largest portion in the

allocation, simply set the mesh size of the triangulation to be some small finite value,  $\delta > 0$ , such that if the absolute measure of a piece is less than  $\delta$ , then each player values it less than  $\varepsilon$ . The flowchart for the various processes that occur to generate the approximate envy-free allocation is shown in Figure 58. The scheme of enquiring the owner of each vertex about its preferred piece is shown as the thick rectangle labeled '1' in Figure 58 part B.



**Figure 58 part A.** Flowchart of approximate envy-free / approximate-fair procedure to allocate resources.

Figure continues in part B below.



**Figure 58 part B.** Flowchart of approximate envy-free / approximate-fair procedure to allocate resources (contd.). The preferred piece can be determined by (1) enquiring the owner of vertex  $V^*$  as per Su [20] or (2) by computing from the owner's marks on the resources

### 8.2.3. *Approximate-fair Procedure for Multiagent Resource Allocation*

In this section, I present an alternative method of applying Sperner's lemma to solve a multiagent resource allocation problems. This idea modifies some parts of the work done by Su, to reduce the costs of agent-mediator communication. The resulting solution is however approximate-fair. The agents are expected to adhere to the following protocol:

#### **Protocol**

If there are  $n$  agents participating, each agent should make  $n-1$  marks on the linear resource, creating  $n$  equal portions of the resource by its valuation.

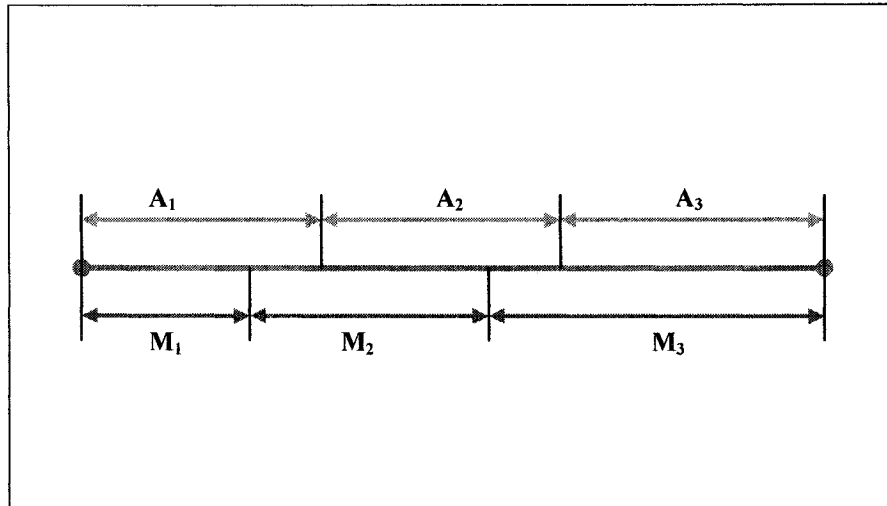
While considering the intervals marked out by each agent, the start and end points of the resource are taken as the first and last mark, respectively. The mediator collects the marks submitted by each agent and uses it to evaluate the appropriate allocation of the resource. If all agents adhere to the protocol, then the procedure can guarantee each agent will be allotted a subset of the portion created by the agent itself. The subset will be smaller by at most  $\epsilon$  than the portion from which it came. The early parts of this procedure are similar to the one put forth by Su. Figure 58 shows the flowchart for the procedure. The procedure can be broken down into two phases:

1. The initialization phase which determines the number of agents,  $n$ , participating in the allocation. The value of grid size  $\delta$ , is also determined. It is used to set the size of the elementary triangles forming the triangulation of the  $n-1$ -simplex.

2. The iterative phase, when the procedure ‘visits’ the vertices of the triangulation. Each vertex  $v^k$  is a “guess” of a feasible approximate-fair solution. If this guess is not approximate-fair to every agent, then label  $v^k$  with label  $j$ ,  $j \in (1, 2, \dots, n)$ . Su’s procedure required that one determine  $j$  by asking the owner of the vertex which piece in the cut-set it prefers. In this procedure, one determines the label as follows: Look up the list of marks submitted by the owner of vertex  $v^k$ . Find a portion in the cut-set that is a superset of one of the pieces marked by the owner. Let this be portion  $j$ . The owner would think that value of portion  $j$  is fair and hence one can set the auxiliary label for vertex  $v^k$  as  $j$ . Next, using Kuhn’s procedure, compute the new value of  $v^k$  (Kuhn’s procedure requires auxiliary labels to compute the new vertex) and repeat step 2.

The following example clarifies the labeling process. Consider 3 agents A, B, and C, that are trying to get an approximate-fair share of the resource. Let vertex  $v^k$  have coordinates (0.22, 0.33, 0.45). If agent A is the owner of the vertex, the mediator pulls up the list of A’s marks. Figure 59 shows the how the portions created by agent A overlap with the mediator’s portions. Notice that  $M_3$  is a superset of  $A_3$  and hence the auxiliary label for vertex  $v^k$  is 3.





**Figure 59.** An example showing how the label of a vertex is determined.  $M_1$ ,  $M_2$ , and  $M_3$  are the coordinates of the vertex at  $(0.22, 0.33, 0.45)$ .  $A_1$ ,  $A_2$ , and  $A_3$  are the owner's marks respectively. In this example, the vertex would be labeled 3 since  $M_3$  is a superset of  $A_3$ .

A couple of points need to be mentioned. The procedure does away with polling the owner of each vertex as it travels through the simplex. This reduces communication costs and agents are free to finish other tasks. Also, the computation time for the approximate fair solution will be less compared to the approximate envy-free solution. This is because, generally, the latency for sending and receiving messages between agents will be much larger than computing the labels locally using the list of agent marks.

Note that the labeling of a vertex requires the presence of at least one piece marked by the agent that is a subset of a piece in the cut-set, determined by the coordinates of vertex  $v^k$ . I present a proof that shows the existence of at least one such agent piece.

Consider an agent and mediator, each of which makes  $n-1$  marks on a linear resource, creating  $n$  portions of a linear resource. Moving from left to right, label the set of agent marks as  $a_1, a_2, \dots, a_{n-1}$ . Similarly the set of mediator marks will be labeled  $m_1, m_2, \dots, m_{n-1}$ . The start and end points of the resource will be labeled  $s$  and  $e$  respectively. The marks delineate intervals as follows:

$A_1 = s - a_1$ , i.e.,  $A_1$  is the interval between the start point ( $s$ ) and the first mark of the agent ( $a_1$ ). Also,  $A_k = a_{k-1} - a_k$  and  $A_n = a_{n-1} - e$ . Similarly, one can write the following for the mediator:  $M_1 = s - m_1$ ,  $M_k = m_{k-1} - m_k$  and finally,  $M_n = m_{n-1} - e$ . Note that  $s$  and  $e$  are the only points that intervals in  $A$  and  $M$  have in common. I assume that no point  $a_i$  coincides with  $m_j$ , for all  $i, j \in (1, 2, \dots, n-1)$ . This assumption is only used to simplify arguments. One can then state the following:

**Theorem 5.** Given two sets of intervals  $(A_1, A_2, \dots, A_n)$  and  $(M_1, M_2, \dots, M_n)$  that partition a linear resource  $R$ , then  $A_i \subset M_j$  for some  $i, j \in (1, 2, \dots, n-1)$ .

That is, there will be at least one interval  $A_i$  that will be a subset of some interval  $M_j$ .

This property is needed in order to guarantee an auxiliary label for every vertex point.

**Proof:** I prove this theorem by contradiction. Let us assume that there exists no interval  $A_i$  that is a subset of some  $M_j$ . What does it mean when one says,  $A_i \subset M_j$ ?

This means that moving from the start to the end point of the resource, one must encounter a sequence of points ordered in the following manner:

$$m_{j-1} - [a_{i-1} - \dots - a_{i-2}] - a_{i-1} - a_i - [a_{i+1} - \dots - a_{i+p}] - m_j$$

The above sequence shows that in order for  $A_i \subset M_j$ , one must find a pair of consecutive m-points that enclose two or more a-points, i.e., one encounters two or more consecutive a-points that do not enclose any m-points, somewhere on the line. Since it is assumed there is no  $A_i \subset M_j$  for some  $i, j \in (1, 2, \dots, n-1)$ , one can conclude that any two consecutive a-points must be necessarily separated by at least one m-point. Since any two consecutive a-points make up an A-interval, it can be concluded that every A-interval should enclose at least one m-point. Create a scoring function  $\Delta$  that denotes how many m-points are enclosed by an interval. Since the entire resource is the interval  $R: \Delta(R)=n-1$ , i.e., the entire resource contains  $n-1$  m-points. Next, compute the value of  $\Delta(A_k)$ . Calculate the value of  $\Delta$  for the first and last A-interval ( $A_1$  and  $A_n$  respectively) separately. Note that the  $s$  is start point of the first M-interval and the first A-interval, i.e.,  $s=a_0=m_0$ . Similarly,  $e=a_n=m_n$ . Thus,  $a_0$  and  $m_0$  coincide and hence it is required that the points should have the following ordering:  $s-m_1-[m_2-m_1]-a_1$ , i.e., there should be at least one m-point before one encounters  $a_1$ . Thus  $\Delta(A_1) \geq 1$ , similarly for the last A-interval one gets:  $\Delta(A_n) \geq 1$ . Since every A-interval in the interior contains at least one m-point one can state:  $\Delta(A_k) \geq 1$  for all  $k \in (1, 2, \dots, n)$ . Summing over all the A intervals one gets,

$$\sum_{k=1}^n \Delta(A_k) \geq n$$

The A-intervals all put together will give the entire resource  $R$ .

$\sum_{k=1}^n A_k = R$  and since  $\Delta(R) = n-1$ , one gets  $\sum_{k=1}^n \Delta(A_k) \neq \Delta(R)$  i.e., the sum of the m-points over all A-intervals is not the same as the number of m-points over the entire

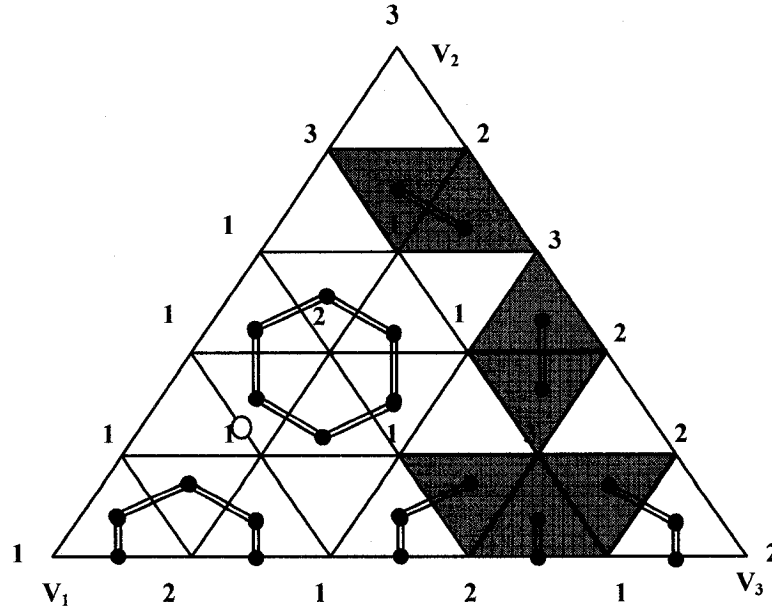
resource  $R$ . This is a contradiction. Therefore the assumption that no  $A_i \subset M_j$  for some  $i, j \in (1, 2, \dots, n-1)$ , is false.  $\square$

### 8.3. Discussion

This section is split into two parts. In the first part, I study the implications of Sperner's lemma in more detail. The second part, "Devising algorithms for exhaustive search" discusses the various issues that were encountered in coming up with a suitable algorithm to locate every fully labeled elementary simplex.

#### 8.3.1. *Improving the Efficiency of Allocation*

Recall that Sperner's lemma states that given a suitably labeled triangulation of an  $n$ -simplex, there will be an odd number of fully labeled elementary simplices. Kuhn's procedure terminates when it finds one such fully labeled elementary simplex (FLES). In fact, Kuhn's procedure cannot reach every FLES in the triangulation. Consider a suitably labeled 2-simplex as shown in Figure 60. I introduce notation in order to explain some concepts.



**Figure 60.** A Sperner labeled triangulation of an 2-simplex. The shaded triangles are the fully labeled elementary simplices,  $\sigma^2$ , and the 1-2 segments are  $\sigma^1$ .

Let  $I^n$  denote the  $n$ -simplex that is triangulated into many smaller elementary  $n$ -simplices  $\tau^n$ . Also every  $\tau^{n-1}$  is a facet of some  $\tau^n$ . If  $\tau^n$  is fully labeled, denote it as  $\sigma^n$ . Each  $\sigma^n$  will have exactly one facet  $\sigma^{n-1}$ . i.e., every fully labeled elementary  $n$ -simplex will have exactly one fully labeled elementary  $n-1$ -simplex as its facet. However  $\sigma^{n-1}$  can also occur as facets of some  $\tau^n$ . Figure 60 shows various possible combinations in which  $\sigma^n$  and  $\sigma^{n-1}$  can exist. Kuhn's procedure can only find those  $\sigma^n$  that are connected to the facet  $I^{n-1}$  ( $I^{n-1}$  forms the "edge" of  $I^n$ ) by a sequence of  $\sigma^{n-1}$ . Starting from a facet of the  $n$ -simplex, it finds each  $\sigma^{n-1}$ , then moves into the interior of  $I^n$  along a path of  $\sigma^{n-1}$ 's. These paths can terminate in one of two ways:

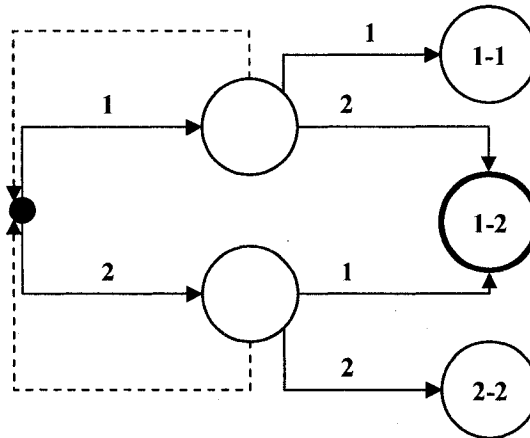
1. The path terminates in  $\sigma^n$ .
2. The path terminates on another  $\sigma^{n-1}$  present on the facet  $I^{n-1}$ .

Notice that there exist  $\sigma^n$ 's in the interior of  $I^n$  that can never be reached by Kuhn's procedures. These  $\sigma^n$ 's come in pairs and are connected to each other by a path of  $\sigma^{n-1}$ 's. Every  $\sigma^n$  represents an alternative allocation that can be approximate fair (or approximate envy-free, based on how the problem is set up). Note that every point in  $I^n$  (including all vertices) represents an allocation of the resource. By definition, if  $x$  is a point in  $I^n$  with  $x=(x_1, x_2, \dots, x_n)$ , then:

$$\sum_{i=1}^n x_i = 1$$

Thus every possible allocation is pareto-efficient. One cannot move from one point to another in  $I^n$  without making at least one agent worse off in the process. Efficiency, however, defined in terms of maximization of social welfare can be different for each allocation  $x$ . The  $\sigma^n$  found by Kuhn's procedure may not be the most efficient. It makes sense therefore to devise a procedure that can find every  $\sigma^n$  in  $I^n$ . Then by comparison, one can find an allocation that is not only approximate-fair (or approximate envy-free), but also the most optimal with respect to the chosen social welfare function.

### 8.3.2. Devising Algorithms for Exhaustive Search



**Figure 61.** NFA for a 1-simplex (line). The desired final state is shown in bold and the  $\lambda$ -transitions are shown by the dotted lines.

A number of issues were encountered while devising a procedure for exhaustive search of  $I^n$ . One can imagine finding  $\sigma^n$  while “traveling” through every  $\tau^n$  in  $I^n$ , as reaching the desired state in a finite state automaton (FSM). What will the automaton for a 1-simplex (line) look like? The facets of the line – which are the 0-simplexes (points) – will represent the events in a FSM. The line segments that triangulate the 1-simplex represent states in the FSM. The NFA for such a procedure is shown in Figure 61.

Next, the NFA for a 2-simplex (triangle) is constructed. Each final state is labeled and the desired final state is circled in bold. The desired final state will have a label 1-2-3, viz.

the labels of a fully labeled elementary simplex. The triangles can have any of the following labels:

1-1-1, 1-1-2, 1-1-3, 1-2-2, **1-2-3**, 1-3-3, 2-2-2, 2-2-3, 2-3-3, 3-3-3.

The final states will therefore also use the above labels. Note that in any  $n$ -simplex, each vertex will be connected to every other vertex by a facet. Hence the ordering of the labels is not important. Thus the following labels are equivalent:  $1-1-2 \leftrightarrow 1-2-1 \leftrightarrow 2-1-1$  and all of them are represented by the state labeled 1-1-2. Figure 62 shows the NFA for the 2-simplex. The events are the elementary  $n-1$ -simplices that form the facets of the elementary  $n$ -simplices. The dotted lines show the  $\lambda$ -transition, which is used to non-deterministically return to the start state. The rationale for the  $\lambda$ -transition is as follows:

In order to determine all the labels of a particular triangle, one needs the labels of at least two of its facets. Every input to the NFA represents one facet of the triangle. Thus two inputs are needed before the labeling of a triangle can be deduced. However, every input to the FSM has two interpretations:

1. It may be the second input for a particular triangle. If so, then the labeled final state denotes the label of the triangle.
2. It may be the first input for a new triangle. If so, the  $\lambda$ -transition takes the FSM to the start state to account for such a “guess.”

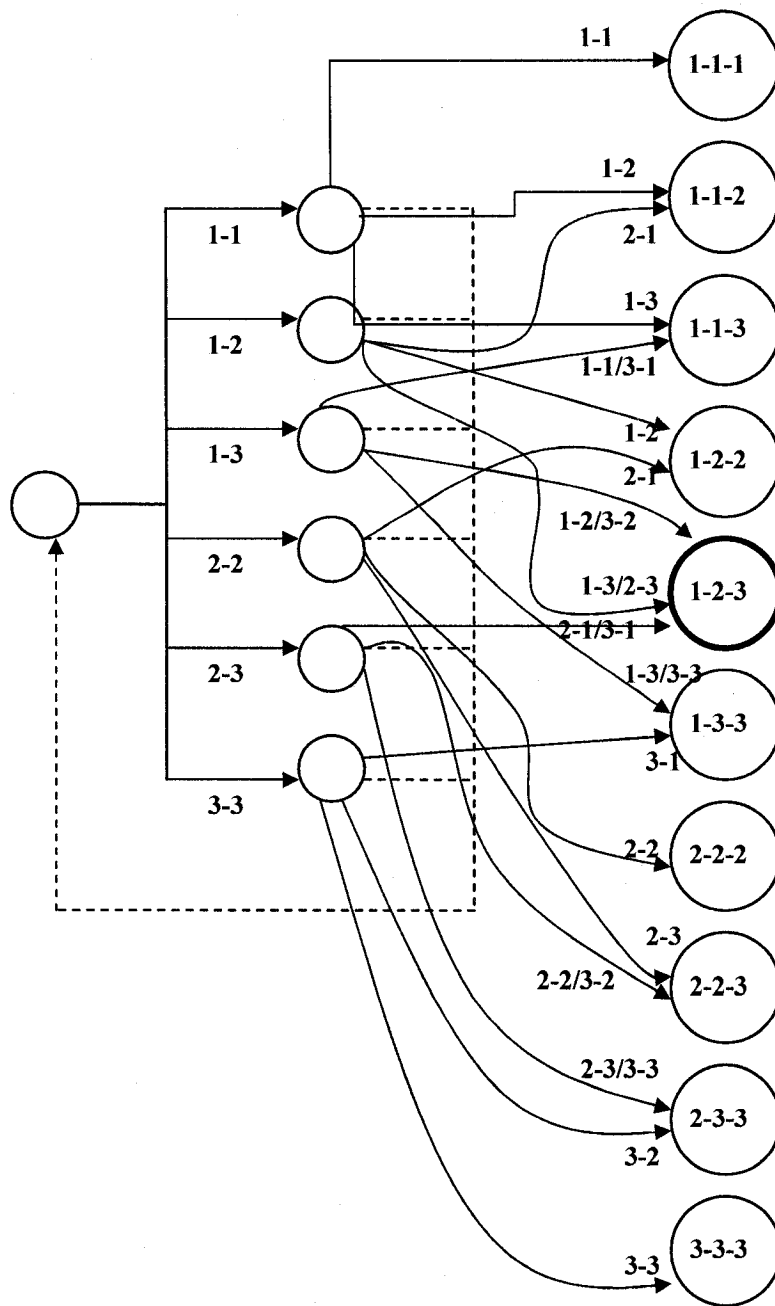
A NFA representation of the FSM is therefore, more natural. Any NFA can be converted to a DFA, but may have exponentially greater states than the NFA in the worst case. It can be observed from Figure 61 and Figure 62 that the complexity of the FSM grows



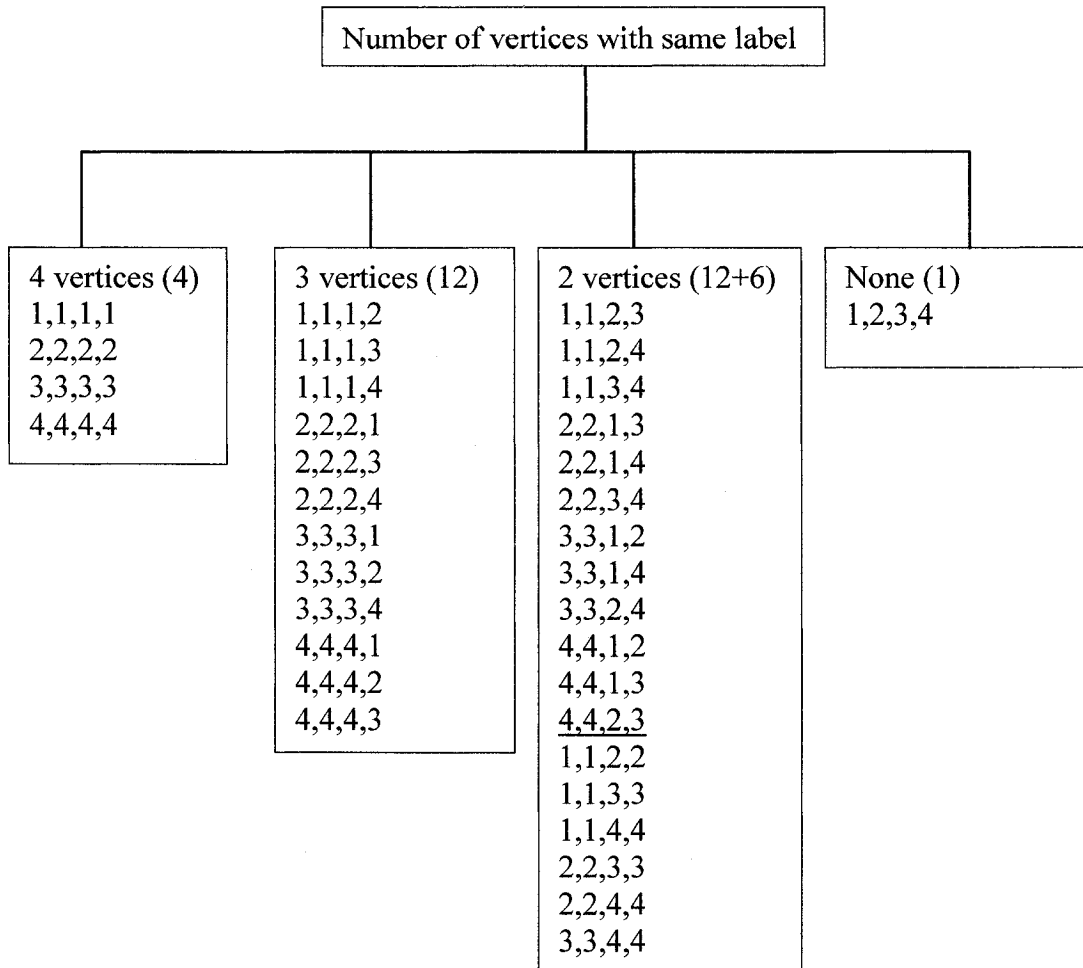
quickly as the dimensionality of the simplex increases. In order to get an idea of the complexity, it is instrumental that the number of labeled states for a generic  $n$ -simplex is determined. I state the sub-problem in the following section.

*8.3.2.1. Sub-problem: How Many Labeled States Exist for an  $n$ -Simplex?*

I proceed by using the 3-simplex (tetrahedron) as an example to understand the issues involved. The 3-simplex has 4 vertices that can be labeled in 35 ways. They are enumerated in Figure 63.



**Figure 62.** The NFA for a 2-simplex (triangle). The labeled states represent the types of labeled elementary simplices. The bold circle shows the FLES and is the “success” state. The dotted line shows the  $\lambda$ -transitions.



**Figure 63.** Computing the combination of labels that can be applied on an elementary 3-simplex. A total of 35 ways are possible.

The labels were generated based on the logic described below. Moving from all 4 vertices having the same label to all of the vertices having distinct labels (i.e., none of the vertices have the same label), the following scenarios arise:

1. 4 vertices have the same label: Thus the four vertices can be replaced with one placeholder and one can label in  ${}^4C_1 = 4$  ways.

2. 3 vertices have the same label: Thus 3 vertices can be replaced with one placeholder. The vertex with the distinct label will get its own placeholder. Since ordering of the labels assigned to placeholders is important, one gets  ${}^4C_1 \times {}^3C_1 = 12$  ways of labeling.
3. 2 vertices have the same placeholder: Thus 2 vertices will be replaced with one placeholder. The following sub-cases occur:
  - 3.1. The other 2 vertices have the same label (They will be different from the first 2 vertices though). These 2 vertices will be replaced with 1 placeholder. Thus 4 labels can put in 2 placeholders in  ${}^4C_2 = 6$  ways. Note that the ordering of the labels is not important in this case.
  - 3.2. The other 2 vertices have distinct labels. Thus there will be 3 placeholders. But because of the way the placeholders were created, one can place 4 labels in 3 placeholders in  ${}^4C_1 \times {}^3C_2 = 4 \times 3 = 12$  ways.
4. None of the vertices have the same label. Thus 4 labels can be placed in 4 placeholders in  ${}^4C_4 = 1$  way.

The formulas for calculating the number of label combinations is based on the specific situation encountered. I explain the logic with the following example:

Consider a particular sequence of numbers that may label a simplex, – 1,1,2,2,2,3,4. Each of the repeated labels can be replaced by a single placeholder, which will give us the following sequence: 1,2,3,4. The underlined placeholder (UP) acts as a stand-in for multiple vertices having the same label. The normal placeholder (NP) is used for each vertex that is distinctly labeled. The ordering of UP s is important. For example:

$\underline{1,2,3,4}=1,1,2,2,2,3$  is different from  $\underline{2,1,3,4}=2,2,1,1,1,3,4$ . So permutations are used while accounting for ordering of UP s. The ordering of NP s is not important. For example:  $\underline{1,2,3,4}=1,1,2,2,2,3,4$  is the same as  $\underline{1,2,4,3}=1,1,2,2,2,4,3$ . Hence combinations are used for NP s. The above sequence of numbers was used to label a 6-simplex (7 vertices). The vertices can “bunch” together in the following ways:

- (7)
- (6,1)
- (5,2) (5,1,1)
- (4,3) (4,2,1) (4,1,1,1)
- (3,3,1) (3,2,2) (3,2,1,1) (3,1,1,1,1)
- (2,2,2,1) (2,2,1,1,1) (2,1,1,1,1,1)
- (1,1,1,1,1,1,1)

Thus there are 15 ways in which the vertices bunch together. This brings us to a sub-problem of the current problem, i.e., a sub-sub-problem.

8.3.2.2. *Sub-sub-problem: Given a number  $n$ , how many distinct sets of numbers can be created, such that the members in each set add up to  $n$ ?*

I previously showed the various sets that are generated when  $n=7$ . I list three different approaches that were attempted to construct a procedure to generate such sets.

Approach 1: Using recursive rules.

I generate the sets for  $n$  using the sequence already generated for  $n-1$ . The following rules are applied to decipher the sequence for  $n$ :

1. The first set in  $n$  is  $(n)$ .
2. Append 1 to every set in  $(n-1)$ .
3. If  $n$  is even, add the set  $(n/2, n/2)$ . If  $n$  is odd, add the set  $((n+1)/2, (n-1)/2)$ . If such a set already exists due to rules 1 and 2, then do not add this set.

Based on the above rules the following sets are generated when  $n=7$ :

n	Rule 1	Rule 2	Rule 3	Skipped sets
1	1	-	-	-
2	(2)	(1,1)	-	-
3	(3)	(2,1)(1,1,1)		-
4	(4)	(3,1) (2,1,1)(1,1,1,1)	(2,2)	-
5	(5)	(4,1)(3,1,1)(2,1,1,1)(1,1,1,1,1)(2,2,1)	(3,2)	-
6	(6)	(5,1)(4,1,1)(3,1,1,1)(2,1,1,1,1)(1,1,1,1,1,1) (2,2,1,1)(3,2,1)	(3,3)	(4,2)(2,2,2)
7	(7)	(6,1)(5,1,1)(4,1,1,1)(3,1,1,1,1)(2,1,1,1,1,1) (1,1,1,1,1,1,1)(2,2,1,1,1)(3,2,1,1)(3,3,1)(2,2,2,1)(4,2,1)	(4,3)	(5,2)(3,2,2)

**Table 6.** Generating number sets using recursive rules. The last column shows the sets that have been missed by the rules.

As can be seen from Table 6 the three rules together are not sufficient to generate all the possible sets. The sets that fail to be generated are mentioned in the “Skipped sets” column. In fact, even if one includes the missing sets of  $n-1$  while generating sets for  $n$ , the rules still miss a few sets of  $n$ . Thus if one generates a sequence for  $n$  just by using the three rules, the number of skipped sets will grow quickly. The three rules are necessary

but not sufficient to generate all the sets. It is clear that more rules need to be defined for generating all the sets. Among the issues that need to be resolved are:

- Is the number of rules bounded?
- How many more rules are needed?
- What are those rules?

Approach 2: Divide the coins

Another approach attempted was what I term “Divide the coins.” Imagine holding  $n$  identical coins that need to be put in various slots. Each slot can hold multiple coins. The number of slots varies from 1 to  $n$ . I try to determine how many different ways the coins can be distributed given the number of slots. Table 7 describes the approach for  $n=8$ .  $B(n)$  is a function that calculates the number of sets that  $n$  will generate. For the example shown in Table 7,  $B(8)=22$ . I do not have an analytical formula for  $B(n)$ .  $F(n,k)$  represents the number of sets that are generated, given the number is  $n$  and  $k$  is the number of slots available. Thus,

$$B(n) = \sum_{k=1}^n F(n,k) \quad (1 \leq k \leq n)$$

From Table 7, it is known that

$$F(n,k) = B(n-k) \quad \text{if } k \geq n/2$$

slots	rem		Number of ways																				
8	0	<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	$F(8,8)=B(0)=1$												
1	1	1	1	1	1	1	1																
7	1	<table border="1"><tr><td>2</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	2	1	1	1	1	1	1	$F(8,7)=B(1)=1$													
2	1	1	1	1	1	1																	
6	2	<table border="1"><tr><td>3</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>2</td><td>2</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	3	1	1	1	1	1	2	2	1	1	1	1	$F(8,6)=B(2)=2$								
3	1	1	1	1	1																		
2	2	1	1	1	1																		
5	3	<table border="1"><tr><td>4</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>3</td><td>2</td><td>1</td><td>1</td><td>1</td></tr><tr><td>2</td><td>2</td><td>2</td><td>1</td><td>1</td></tr></table>	4	1	1	1	1	3	2	1	1	1	2	2	2	1	1	$F(8,5)=B(3)=3$					
4	1	1	1	1																			
3	2	1	1	1																			
2	2	2	1	1																			
4	4	<table border="1"><tr><td>5</td><td>1</td><td>1</td><td>1</td></tr><tr><td>4</td><td>2</td><td>1</td><td>1</td></tr><tr><td>3</td><td>3</td><td>1</td><td>1</td></tr><tr><td>3</td><td>2</td><td>2</td><td>1</td></tr><tr><td>2</td><td>2</td><td>2</td><td>2</td></tr></table>	5	1	1	1	4	2	1	1	3	3	1	1	3	2	2	1	2	2	2	2	$F(8,4)=B(4)=5$
5	1	1	1																				
4	2	1	1																				
3	3	1	1																				
3	2	2	1																				
2	2	2	2																				
3	5	<table border="1"><tr><td>6</td><td>1</td><td>1</td></tr><tr><td>5</td><td>2</td><td>1</td></tr><tr><td>4</td><td>3</td><td>1</td></tr><tr><td>4</td><td>2</td><td>2</td></tr><tr><td>3</td><td>3</td><td>2</td></tr></table>	6	1	1	5	2	1	4	3	1	4	2	2	3	3	2	$F(8,3)=B(5)=5$					
6	1	1																					
5	2	1																					
4	3	1																					
4	2	2																					
3	3	2																					
2	6	<table border="1"><tr><td>7</td><td>1</td></tr><tr><td>6</td><td>2</td></tr><tr><td>5</td><td>3</td></tr><tr><td>4</td><td>4</td></tr></table>	7	1	6	2	5	3	4	4	$F(8,2)=B(6)=n/2=4$												
7	1																						
6	2																						
5	3																						
4	4																						
1	7	<table border="1"><tr><td>8</td></tr></table>	8	$F(8,1)=B(7)=1$ (for all n)																			
8																							

**Table 7.** Generating number sets by determining the combinations in which the coins can be put in slots.

When the number of slots available is 2 or 3, the function  $F(n,k)$  is unknown.

Also

$$F(n,1) = 1 \quad \forall n$$

and

$$F(n,2) = \frac{\lfloor n \rfloor}{2} \quad \forall n$$



But the analytic function is unknown for  $2 < k < n/2$ . Hence  $B(n)$  cannot be computed.

Approach 3: Generation of sets through trial and error.

The third approach was to attempt to create a program that would generate all the sets for a given number. If this was possible then one could simply count the sets generated for each  $n$  and then try to fit an equation onto the graph of  $n$  vs.  $B(n)$ . Using recursive functions calls I was able to create such a procedure. The code, written in Java, is shown below:

```

import java.io.*;
public class genSets{
int count = 1;
public static void main(String[] args){
int n = Integer.parseInt(args[0]);
String str = "";
genSets gs = new genSets();
gs.comp(str,n,n,0,true);
}
public void comp(String prefix,int base_n,int n, int rem_n,boolean
finalRes){
if(rem_n>n){
String tempPrefix = prefix;
prefix = prefix+", "+n;
comp(prefix,base_n-n,n,rem_n-n,false);
if(n==1);
else{
n=n-1;
rem_n=base_n-n;
comp(tempPrefix,base_n,n,rem_n,false);
}}
else{
String tempPrefix = prefix;
if(rem_n>0){
System.out.println(prefix+", "+n+", "+rem_n);
count++;
prefix = prefix+", "+n;
comp(prefix,base_n-n,rem_n,0,false);
if(n==1);
else{
n=n-1;
rem_n=base_n-n;
comp(tempPrefix,base_n,n,rem_n,false);
}}
else{ //rem_n==0
if(n==1) ;
else{ //n>1
n=n-1;
rem_n=base_n-n;
comp(tempPrefix,base_n,n,rem_n,false);
}}}}
}
}
}
}
}

```

The output of the program for  $n=8$  is shown in Figure 64.

```
C:\WINDOWS\system32\cmd.exe
C:\Documents and Settings\iyerk\My Documents\Research>java genSets 8
.7,1
.6,2
.6,1,1
.5,3
.5,2,1
.5,1,1,1
.4,4
.4,3,1
.4,2,2
.4,2,1,1
.4,1,1,1,1
.3,3,2
.3,3,1,1
.3,2,2,1
.3,2,1,1,1
.3,1,1,1,1,1
.2,2,2,2
.2,2,2,1,1
.2,2,1,1,1,1
.2,1,1,1,1,1,1
.1,1,1,1,1,1,1,1
C:\Documents and Settings\iyerk\My Documents\Research>
```

**Figure 64.** The number sets generated by the java program based on the trial and error approach.

The algorithm is able to exhaustively generate all sets. I then used the algorithm to find the values of  $B(n)$  for values of  $n$  from 1 to 100. The computation time was roughly 4 hrs on a 1.2 GHZ Celeron processor with 256 MB RAM. Table 8 shows the values of  $n$  vs.  $B(n)$  and Figure 65 shows the graph of the rise of  $B(n)$  with respect to  $n$ .

n	B(n)
1	1
2	2
3	3
4	5
5	7
6	11
7	15
8	22
9	30
10	42
11	56
12	77
13	101
14	135
15	176
16	231
17	297
18	385
19	490
20	627
21	792
22	1002
23	1255
24	1575
25	1958
26	2436
27	3010
28	3718
29	4565
30	5604
31	6842
32	8349
33	10143
34	12310
35	14883
36	17977
37	21637
38	26015
39	31185
40	37338
41	44583
42	53174
43	63261
44	75175
45	89134

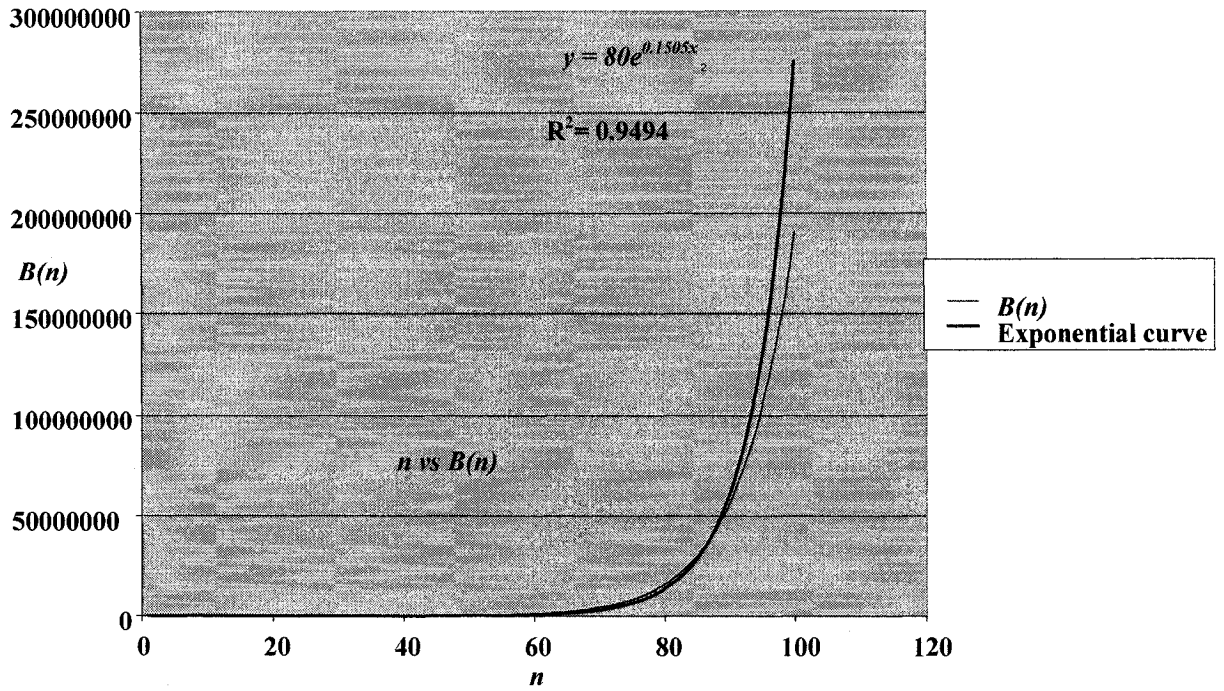
<b>n</b>	<b>B(n)</b>
46	105558
47	124754
48	147273
49	173525
50	204226
51	239943
52	281589
53	329931
54	386155
55	451276
56	526823
57	614154
58	715220
59	831820
60	966467
61	1121505
62	1300156
63	1505499
64	1741630
65	2012558
66	2323520
67	2679689
68	3087735
69	3554345
70	4087968
71	4697205
72	5392783
73	6185689
74	7089500
75	8118264
76	9289091
77	10619863
78	12132164
79	13848650
80	15796476
81	18004327
82	20506255
83	23338469
84	26543660
85	30167357
86	34262962
87	38887673
88	44108109
89	49995925
90	56634173
91	64112359

n	B(n)
92	72533807
93	82010177
94	92669720
95	104651419
96	118114304
97	133230930
98	150198136
99	169229875
100	190569292

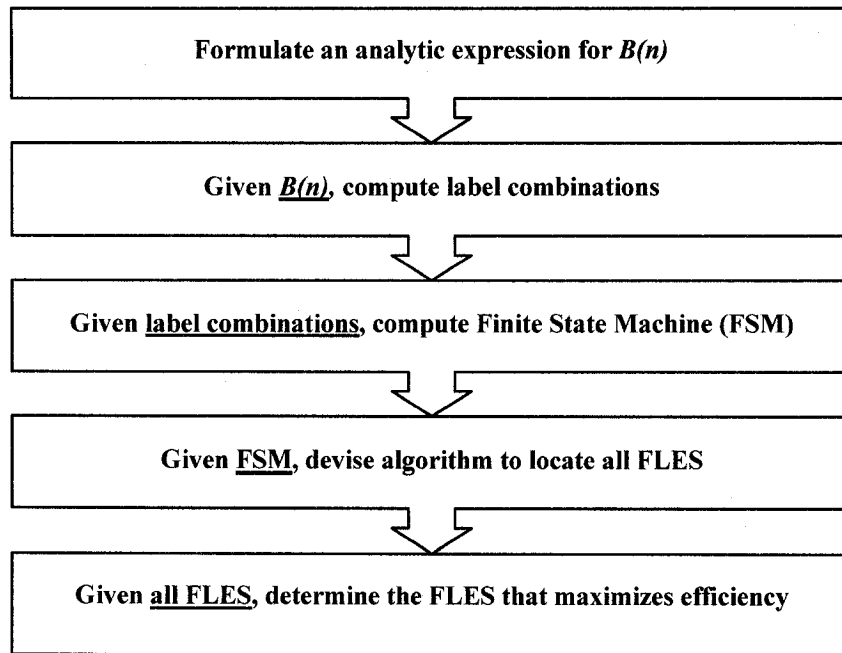
**Table 8.** Values of  $B(n)$  for  $1 \leq n \leq 100$  generated by the java program.

The curve of  $B(n)$  is be fitted with an exponential curve with an R-squared value of 0.9494. Curve fitting with other types of curves like the polynomial or power-law type curves gives poorer results.

Finally, I present a “trackback” (Figure 66) to give a perspective of the issues that need to be resolved in order to allow for efficient allocation of a resource.



**Figure 65.** The relationship of  $B(n)$  vs.  $n$ . An exponential curve with the values shown in the figure gives the best fit.



**Figure 66.** A “trackback” that shows the dependency of the various problems that need to be solved before an algorithm can be devised to do an exhaustive search of a triangulated  $n$ -simplex

#### 8.4. Conclusion

This chapter discusses the applicability of Sperner’s lemma for solving multiagent resource allocation problems. First the lemma was explored, whereby the concepts of triangulation and Sperner labeling are explained. After stating the lemma as applicable to any  $n$ -simplex, I explain how Kuhn’s procedure helps in locating a fully labeled elementary simplex. It is then shown that a resource allocation problem can be mapped onto an appropriately labeled triangulation of an  $n$ -simplex. An approximate envy-free allocation mechanism as proposed by Su is mentioned first. Using appropriate conventions and assumptions, it is shown how one can apply the mathematical result to



solve a multiagent resource allocation problem. Next, an allocation mechanism is proposed that has better communication efficiency than Su's procedure. However this result is only approximate fair (which is a weaker condition than approximate envy-free). Every fully labeled elementary simplex is efficient in the pareto-optimal sense. If efficiency is studied in terms of social welfare however, one elementary simplex may represent a more efficient allocation than another. Kuhn's procedure does not locate every possible fully labeled elementary simplex. Constructing a procedure that locates every fully labeled elementary simplex is however non trivial. Modeling the procedure as a finite state machine, it was observed that the number of final states grows quickly as the number,  $n$ , of agents increases. The number of final states depends on the combination of labels that can be applied to an elementary simplex. This in turn depends on the number of ways the labels can "bunch" together. I present three approaches to solve these problems. Unfortunately, none of them are sufficient to completely solve the problems. I believe that the simple yet powerful combinatorial property of Sperner's lemma can be adapted for solving multiagent resource allocation problems elegantly, but the many open problems discussed in this chapter need to be solved first.

# Chapter 9

## Conclusion

Software agents are increasingly being used to conduct commerce and trading on the Internet. It therefore becomes essential to focus on the design of negotiation protocols and procedures that enable fair and unbiased allocation of scarce resources. This dissertation presents procedures and protocols for the allocation of hyperdimensional resources in a multiagent system. The resources are hyperdimensional, because they can have several dimensions that are not necessarily linear. The agents contending for the resources are assumed to be selfish, autonomous and capable of lying. It is also assumed that each agent is entitled to  $1/n$  of the entire resource. The resource is assumed to be scarce and each agent's goal is to obtain as much of the resource as possible. For the protocols and procedures that are presented in this dissertation, the resource is required to be infinitely divisible. If the resource is recombining as well, then wastage incurred during allocation is reduced. Recombinability of a resource is not required for the procedures to apply, but it is preferred.

Ideas from various fields are analyzed to come up with a negotiation protocol that enables allocation of resources among agents while satisfying various desirable properties. The

literature survey covers existing research from disparate areas including mathematics, economics, multiagent systems, social justice, and social welfare.

As part of my original research contribution, an extensive list of criteria for negotiation is presented. This has been culled from existing literature. In addition, three new criteria have been proposed: verifiability, dimensionality, and topology. My intention has been to analyze existing protocols exhaustively as well enable the design of new protocols from the ground up. It is therefore suggested that this work be used as a starting point for protocol engineering.

Second, protocols and procedures are presented for the allocation of one-dimensional linear and circular resources. Third, the work is extended to account for two-dimensional planar resources such as land. Fourth, protocols and procedures are presented for two-dimensional cylindrical resource allocation. The protocols require that agents mark out their regions of interest as rectangular shaped portions only. This enables allocation of portions that are well-shaped. It also serves in simplifying analysis of the problem. Two conditions; called Iyer's conditions, are put forth in order to determine if a solution exists. If one or more of these conditions are satisfied, then appropriate procedures can come up with a fair allocation. In case the overlap degree condition is true, then a procedure converts the overlap of agent rectangles into a corresponding graph. A hill-climbing algorithm applied to this graph finds a suitable allocation of the resource.

The two dimensional procedures are analyzed in detail for their benefits and drawbacks. A two-dimensional resource allocation procedure is more efficient than traditional one-dimensional resource allocation procedures (if some portions of the resource have negative utilities). However they qualitatively differ from one-dimensional procedures. For the case of the one-dimensional resource allocation procedures, if the agents followed the protocol, a fair allocation of the resource is guaranteed. However for the two-dimensional case, simply following the protocol does not guarantee that a feasible solution exists. Hence Iyer's conditions are specified that check the existence of a feasible allocation. The relationship of the non-overlap condition and the overlap degree condition is studied. It is shown how they partially overlap each other.

In addition, the proposed protocols are analyzed in detail for fitness with respect to the various criteria compiled in this dissertation. The proposed procedures are fair, unbiased, and verifiable. They account for the topology and the dimensionality of the resource. The computational cost is of the order of  $O(n^2)$ . The procedures are strategy-proof, private, anonymous, and make no assumptions about the utility functions of agents. However the procedures are inefficient.

Finally, the applicability of Sperner's lemma to multiagent resource allocation is studied. Existing work that proposes an approximate envy-free scheme to allocate a resource to  $n$  agents using the property of Sperner's lemma is explained first. Then an alternative scheme that improves communication efficiency is presented. This is however only approximate fair. Attempts were made to come up with procedures that improve the

efficiency of an allocation. Efficiency here is meant in the sense of maximizing social welfare. Constructing such a procedure was however non-trivial. The various issues that need to be resolved for construction of the procedure are highlighted.

## References

- [1] M. Wooldridge, Introduction to MultiAgent Systems, John Wiley & Sons, Chichester, England, 2002.
- [2] M.P. Singh, and M.N. Huhns, Service-Oriented Computing: Semantics, Processes, Agents, John Wiley & Sons, West Sussex, England, 2005.
- [3] FIPA ACL Message Structure Specification, Foundation for Intelligent Physical Agents, 2002.
- [4] G. Weiss, (Ed.), Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, The MIT Press, Cambridge, MA, 2000.
- [5] J.O. Kephart, and A.R. Greenwald, Shopbot Economics. Autonomous Agents and Multi-Agent Systems 5 (2002) 255-287.
- [6] M. Huhns, Agents as Web services. IEEE Internet Computing 6 (Jul/Aug 2002) 93-95.
- [7] J.S. Rosenschein, and G. Zlotkin, Rules of Encounter: Designing Conventions for Automated Negotiation among Computers, MIT Press, London, England, 1994.
- [8] R. Davis, and R.G. Smith, Negotiation Distributed as a Metaphor for Problem Solving. Artificial Intelligence 20 (January 1983) 63-109.
- [9] M.P. Singh, Agent Communication Languages: Rethinking the Principles. IEEE Computer 31 (1998) 40-47.
- [10] V. Tammaa, S. Phelps, I. Dickinson, and M. Wooldridge, Ontologies for supporting negotiation in e-commerce. Engineering applications of artificial intelligence 18 (2005) 223-236.

- [11] W. Vickrey, Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance* 16 (1961) 8-37.
- [12] R. Martin, The St. Petersburg Paradox. in: E.N. Zalta, (Ed.), *The Stanford Encyclopedia of Philosophy*, Fall 2004.
- [13] D.E. Bell, H. Raiffa, and A. Tversky, *Decision Making: Descriptive, Normative and Prescriptive Interactions*, Cambridge University Press, New York, NY, 1988.
- [14] S.J. Brams, and A.D. Taylor, *Fair Division: From Cake-Cutting to Dispute Resolution*, Cambridge University Press, New York, 1996.
- [15] J. Harrington, J.G.A. Pocock, R. Geuss, and Q. Skinner, *Harrington: 'The Commonwealth of Oceana' and 'A System of Politics'*, Cambridge University Press, 1992.
- [16] I. Peterson, Formulas for fairness, *Science News*, May, 1996.
- [17] D.B.H. Denoon, and S.J. Brams, Fair Division: A New Approach to the Spratly Islands Controversy. *International Negotiation* 2 (February 1997) 303-329.
- [18] C.M. Crawford, Internet online backup system provides remote storage for customers using IDs and passwords, Jun 23, 1998.
- [19] M. Gardner, *Aha! Insight*, W.H. Freeman & Company, New York, 1978.
- [20] F.E. Su, Rental harmony: Sperner's lemma in fair division. *American Mathematical Monthly* 106 (1999) 930-942.
- [21] B. Ginsberg, Gerrymander, *Microsoft Encarta Online Encyclopedia*, 2006.
- [22] B. Hayes, Machine Politics, *American Scientist*, Dec 1996, pp. 522-526.
- [23] E.H. Durfee, and V.R. Lesser, Using partial global plans to coordinate distributed problem solvers, *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, Milan, 1987, pp. 875-883.

- [24] C.H. Brooks, and E.H. Durfee, Congregation Formation in Multiagent Systems. *Autonomous Agents and Multi-Agent Systems* 7 (Jul 2003) 145-170.
- [25] C. Li, J. Giampapa, and K. Sycara, Bilateral Negotiation Decisions with Uncertain Dynamic Outside Options. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Special Issue on Game-theoretic Analysis and Stochastic Simulation of Negotiation Agents* 36 (January 2006).
- [26] S. Kraus, and J. Wilkenfeld, Negotiations over time in a multiagent environment, *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence, Sydney, August 1991*, pp. 56-61.
- [27] R. Lin, S. Kraus, J. Wilkenfeld, and J. Barry, An Automated Agent for Bilateral Negotiation with Bounded Rational Agents with Incomplete Information, *Proc. of ECAI, 2006*.
- [28] S.S. Fatima, M. Wooldridge, and N.R. Jennings, Multi-Issue Negotiation with Deadlines. *Journal of Artificial Intelligence Research* 27 (2006) 381-417.
- [29] S. Kraus, J.S. Rosenschein, and M. Fenster, Exploiting Focal Points Among Alternative Solutions: Two Approaches. *Annals of Mathematics and Artificial Intelligence* 28 187 - 258.
- [30] F. Tohme, Negotiation as a Resource Allocation Process, Washington University in St. Louis, Department of Computer Science and Engineering, 1996, pp. 51.
- [31] S. Krumke, M. Lipmann, W.E.d. Paepe, D. Poensgen, J. Rambau, L. Stougie, and G.J. Woeginger, How to cut a cake almost fairly, *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms, 2002*, pp. 263-264.



- [32] S. Even, and A. Paz, A note on cake cutting. *Discrete Applied Mathematics* 7 (1984) 285-296.
- [33] W.A. Webb, How to cut a cake fairly using a minimal number of cuts. *Discrete Applied Mathematics* 74 (April 1997) 183-190.
- [34] E. Peterson, and F.E. Su, Four-Person Envy-Free Chore Division, *Mathematics Magazine*, 2002, pp. 117-122.
- [35] W. Stromquist, How to Cut a Cake Fairly. *American Mathematical Monthly* 87 (October 1980) 640-644.
- [36] A.K. Austin, Sharing a cake. *Mathematical Gazette* 66 (Oct 1982) 212-215.
- [37] S.J. Brams, A.D. Taylor, and W.S. Zwicker, A Moving-Knife Solution to the Four-Person Envy-Free Cake-Division Problem. *Proceedings of the American Mathematical Society* 125 (Feb 1997) 547-554.
- [38] M. Magdon-Ismail, C. Busch, and M. Krishnamoorthy, Cake-Cutting is Not A Piece of Cake, *Proceedings of the 20th International Symposium on Theoretical aspects of Computer Science* 2003.
- [39] J. Sgall, and G.J. Woeginger, A Lower Bound for Cake Cutting, *Algorithms - ESA* 2003, 2003, pp. 459-469.
- [40] G.J. Woeginger, and J. Sgall, *On the complexity of cake cutting*, ITI Series, Charles University, Prague, 2004.
- [41] J. Edmonds, and K. Pruhs, Cake cutting really is not a piece of cake, *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, Miami, Florida, 2006, pp. 271-278.
- [42] T.P. Hill, Sharing a cake. *Mathematical Gazette* 66 (Oct 1982) 212-215.

- [43] L.E. Dubins, and E.H. Spanier, How to cut a cake fairly. *American Mathematical Monthly* 68 (Jan 1961) 1-17.
- [44] A. Beck, Constructing a Fair Border. *American Mathematical Monthly* 94 (Feb 1987) 157-162.
- [45] W.A. Webb, A Combinatorial Algorithm to Establish a Fair Border. *European Journal of Combinatorics* 11 (May 1990) 301-304.
- [46] F. Maccheroni, and M. Marinacci, How to cut a pizza fairly: Fair division with decreasing marginal evaluations. *Social Choice and Welfare* 20 (2003) 457-465.
- [47] R.G. Bartle, *The Elements of Integration and Lebesgue Measure*, Wiley-Interscience, New York, 1995.
- [48] W. Thomson, Children crying at birthday parties. Why? Fairness and incentives for cake division problems, Working Papers from University of Rochester - Center for Economic Research (RCER), Oct 2005.
- [49] K. Iyer, and M. Huhns, Multiagent Negotiation for Fair and Unbiased Resource Allocation. in: R. Meersman, and Z. Tari, (Eds.), *OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2005*, Springer, Oct 2005, pp. 453-465.
- [50] C.P. Chambers, Allocation rules for land division. *Journal of Economic Theory* 121 (2005) 236-258.
- [51] J.S. Mill, and G. Sher, *Utilitarianism*, Hackett Publishing Company, 2002.
- [52] Y. Chevaleyre, P.E. Dunne, U. Endriss, J. Lang, M. Lemaitre, N. Maudet, J. Padget, S. Phelps, J.A. Rodriguez-Aguilar, and P. Sousa, Issues in Multiagent Resource Allocation. *Informatica* 30 (2006) 3-31.

- [53] T. Sandholm, Contract types for satisficing task allocation: I theoretical results., AAAI Spring Symposium Series: Satisficing Models, Stanford University, CA, 1998, pp. 68-75.
- [54] S.J. Brams, Negotiation Games, Routledge, New York, 2003.
- [55] S.J. Brams, D.M. Kilgour, and S. Merrill, Arbitration Procedures. in: H.P. Young, (Ed.), Negotiation Analysis, University of Michigan Press, Ann Arbor, 1991, pp. 47-65.
- [56] D. Gale, Mathematical Entertainments. Mathematical Intelligencer 15 (1993) 48-52.
- [57] J.B. Barbanel, and A.D. Taylor, Preference relations and measures in the context of fair division. Proceedings of the American Mathematical Society 123 (1995) 2061-2070.
- [58] H.P. Young, Equity: In Theory and Practice, Princeton University Press, Princeton, NJ, 1994.
- [59] V. Crawford, A Game of Fair Division. The Review of Economic Studies 44 (1977) 235-247.
- [60] V. Crawford, A Procedure for Generating Pareto-Efficient Egalitarian-Equivalent Allocations. Econometrica 47 (1979) 49-60.
- [61] G. Demange, Implementing Efficient Egalitarian Equivalent Allocations. Econometrica 52 (1984) 1167-1178.
- [62] W. Stromquist, and D.R. Woodhall, Sets on which several measures agree. Journal of Mathematical Analysis and Applications 108 (1985) 241-248.
- [63] H. Steinhaus, The problem of fair division. Econometrica 16 (1948) 101-104.
- [64] S.J. Brams, and T.R. Kaplan, Dividing the indivisible. Journal of Theoretical Politics 16 (2004) 143-173.

- [65] S.J. Brams, and D.M. Kilgour, Competitive Fair Division. *Journal of Political Economy* 109 (2001) 418-443.
- [66] G. Schneider, and U.S. Kramer, The limitations of fair division: an experimental evaluation of three procedures. *Journal of Conflict Resolution* 48 (Aug 2004) 506-524.
- [67] H.W. Kuhn, On Games of Fair Division. in: M. Shubik, (Ed.), *Essays in Mathematical Economics in honor of Oskar Morgenstern*, Princeton University Press, Princeton, NJ, 1967, pp. 29-37.
- [68] H. Steinhaus, and others, A note on the ham sandwich theorem. *Mathesis Polska* 9 (1938) 26-38.
- [69] A.M. Fink, A Note on the Fair Division Problem. *Mathematics Magazine* 37 (1964) 341-342.
- [70] D.R. Woodall, A Note on the Cake-Division Problem. *Journal of Combinatorial Theory, Series A* 42 (1986) 300-301.
- [71] M. Berliant, W. Thomson, and K. Dunz, On the fair division of a heterogeneous commodity. *Journal of Mathematical Economics* 21 (1992) 201-206.
- [72] S.J. Brams, and A.D. Taylor, *The Win-Win Solution: Guaranteeing Fair Shares to Everybody*, W. W. Norton & Company, 1999.
- [73] I. Bezáková, and V. Dani, Allocating indivisible goods. *ACM SIGecom Exchanges* 5 (April 2005) 11-18.
- [74] A. Asadpour, and A. Saberi, An Approximation Algorithm for Max-min Fair Allocation of Indivisible Goods, *STOC 2007*, 2007.
- [75] A. Kumar, and J. Kleinberg, Fairness measures for resource allocation. *focs 00* (2000) 75-85.

- [76] S.J. Brams, and D.L. King, Efficient fair division: Help the worst off or avoid envy? *Rationality and Society* 17 (2005) 387-421.
- [77] Y. Dutil, Social Choice And Equity Theories: Seeking The Common Good As A Common Ground, 55th International Astronautical Congress, Vancouver; Canada, 2004, pp. 1-8.
- [78] H. Gersbach, Dividing resources by flexible majority rules. *Social Choice and Welfare* 23 (Oct 2004) 295-308.
- [79] S.J. Brams, M.A. Jones, and C. Klamler, Better Ways to Cut a Cake. *Notices of the AMS* 53 (2006) 1314-1321.
- [80] M.G. Raith, and F. su, Procedural support for cooperative negotiations: theory and implementation. in: K.J. Engemann, and G.E. Lasker, (Eds.), *Advances in Decision Technology and Intelligent Information Systems*, The International Institute for Advanced Studies in Systems Research and Cybernetics, Windsor, Canada, 2000, pp. 31-36.
- [81] C.-J. Haake, M.G. Raith, and F.E. Su, Bidding for envy-freeness: A procedural. *Social Choice and Welfare* 19 (2002) 723-749.
- [82] C. Arnsperger, Envy-Freeness and Distributive Justice. *Journal of Economic Surveys* 8 (1994) 155-186.
- [83] P.C. Fishburn, and R.K. Sarin, Fairness and Social Risk I: Unaggregated Analyses. *Management Science* 40 (1994) 1174-1 188.
- [84] D.P. Bertsekas, and Gallager, *Data Networks*, Prentice Hall, 1992.
- [85] E.A. Pazner, and D. Schmeidler, Egalitarian Equivalent Allocations: A New Concept of Economic Equity. *Quarterly Journal of Economics* 92 (Nov 1978) 671-687.

- [86] J. Robertson, and W. Webb, *Cake Cutting Algorithms*, A K Peters Ltd, Nattick, MA, 1998.
- [87] U. Endriss, and N. Maudet, On the Communication Complexity of Multilateral Trading: Extended Report. *Journal of Autonomous Agents and Multi-Agent Systems* 11 (2005) 91-107.
- [88] S. Kraus, J. Wilkenfeld, and G. Zlotkin, Multiagent negotiation under time constraints. *Artificial Intelligence* 75 (1995) 297-345.
- [89] I. Stewart, Your Half 's Bigger Than My Half!. *Scientific American*, December 1998, pp. 112-114.
- [90] M.N. Huhns, and A.K. Malhotra, Negotiating for Goods and Services. *IEEE Internet Computing* 3 (July 1999) 97-99.
- [91] J.K. Goeree, C.A. Holt, and J.O. Ledyard, Report On Results Of Economic Experiments Examining Performance Properties Of Simultaneous Multiple Round Spectrum License Auctions With And Without Combinatorial Bidding, Federal Communications Commission, July 2006.
- [92] M. Henle, *A Combinatorial Introduction to Topology*, Dover Publications, Mineola, NY, 1994.
- [93] S.J. Russell, and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, NJ, 2002.
- [94] E. Sperner, Neuer Beweis für die Invarianz der Dimensionszahl und des Gebietes. *Abh. Math. Sem. Hamburg. Univ.* 6 (1928) 265-272.
- [95] H.W. Kuhn, Simplicial Approximation of Fixed Points. *Proc. Nat. Acad. Sci. U.S.A.* 61 (1968) 1238-1242.

[96] D.I.A. Cohen, On the Sperner lemma. J. Combin. Theory 2 (1967) 585-587.