2011

AN EQUIVALENT CLASS SOLUTION FOR A COMPLETE TEST OF A PARALLEL PROGRAM

Peter Michael Wolf
The University of Montana

AN EQUIVALENT CLASS SOLUTION FOR A COMPLETE TEST OF A PARALLEL PROGRAM

By

PETER MICHAEL WOLF

Bachelors of Science, The University of Montana, Missoula, Montana, 2009

Thesis

presented in partial fulfillment of the requirements
for the degree of

Masters of Science
in Computer Science

The University of Montana
Missoula, MT

May 2011

Approved by:

Stephen Sprang, Associate Provost for Graduate Education
Graduate School

Joel Henry, Chair
Computer Science

Doug Raiford
Computer Science

George McRae
Mathematical Sciences

COPYRIGHT NOTICE

Wolf, Peter, M.S., Spring 2009 Computer Science

Solving the Scheduling Problem for Parallel Programs

Chairperson: Joel Henry

With the advent of multi-core chips (multiple CPUs on a single chip) a fundamental shift in the design of programs is taking place. Previous use of parallel code in programs was limited to servers that contained multiple CPUs on separate chips. This idea is starting to change, with the majority of chips sold today containing multiple CPUs which require parallel code in software programs.

The use of parallel code is not without problems. Non-parallel code executes programming statements in the same order, every time the same inputs are used. This result contrasts parallel code in the extreme. Identical inputs and conditions in no way to guarantee that parallel code will result in the same order of execution. For that reason testing parallel code is dramatically more difficult. It is because of this difficulty that I propose a solution that will allow a tester to guarantee a complete coverage of the shared memory parallel code through a series of non-exhaustive tests.

This solution relies on the fact that the majority of the lines of code in the different threads don't access the shared memory of program. This concept is what allows the creation of equivalent execution classes. Two execution schedules are equivalent if the critical sections that conflict with previously executed sections are in the same order in the schedule and the grouping of code between these sections are simply permutations of legal execution ordering.

Contents

Introduction 1

Literature Review 4

Implementation Background 12

Algorithm Design 19

Parsing Algorithm Design 21

Schedule Finding Algorithm Design 23

Large Scale Example 25

Experiments Set Up 31

Experiments 34

Case 1 36

Case 2 36

Case 3 37

Case 4 39

Case 5 41

Case 6 42

Results 43

Conclusion 47

Future Work 50

Works Cited 52

Appendix A 56

Appendix B 58

Appendix C 60

Appendix D 62

Appendix E 65

Appendix F 79

Appendix G 85

Appendix H 98

Appendix I 100

Appendix J 155

Appendix K 159

Appendix L 169

Introduction

With the advent of multi-core CPUs a fundamental design shift is now taking place in the world of Computer Science. , Traditionally the primary design of software was through the use of serial programing (programs executed commands in a sequence determined by the programmer). However, with a majority of consumer grade chips in the past few years containing a two or more CPUs (multi-core chips) the concept of parallel code is necessary to take advantage of the power these chips provide. In the past a majority of the parallel code used was limited to servers that contained at least two CPUs, each on a separate chip. This fact limited the number of programs that utilized parallel coding. With the shift to multi-core chips the number of programs that can take advantage of parallel code has dramatically increased.

This increasing need for parallel code has shown that simply parallelizing pre-existing serial code does not work well and proves difficult to test. There are fundamental differences in the way parallel code works that makes the standard testing practices invalid. A tester may test each thread of a parallel program for correctness but when the threads are interleaved during execution unintended results can develop. One type of unintended results is commonly called *Data Race Conditions*. These conditions are comprised of different sequences of events that result in invalid values in the shared memory.

These type of conditions are hard to detect because of the way parallel code works. Every time a serial program is run the order of execution is the same but when a parallel program is run the order in which the different threads execute can vary. This variation is the results of the processor scheduling the lines of code differently. The amount of

variation can be small to large depending if the program interacts with external input, depends on specific conditions or how other programs are executed. Depending on the cause of the data race condition it is possible that it will only occur under a very specific schedule.

It is for this very reason that a new approach must be taken with regard to the way parallel code is tested. It is no longer valid to only thoroughly test the individual threads; one must also take into account the way the different threads are scheduled. That concept makes testing parallel code extremely complex and potentially intractable given the explosion of permutations possible.

As a rough estimate the number of possible schedules is $T\hat{}(K-1)$, where T is the number of threads in the program plus an additional section to represent the global section of the program, and K is the number of lines of code in the program. While the true number of schedules is typically much less this does show that the growth rate is exponential. This means that even for a program with a limited number of threads the number of possible schedules can still be extremely high.

It is for this reason that a new approach had to be designed to solve this problem. From the solution presented here arose from observation of the fact that in the threads of a parallel program only a small percentage of code accesses the shared memory. For the remainder of this paper a program is the code that threads will execute. This means that a large percentage of the code can be considered non-critical. For the remainder of this paper non-critical blocks of code indicate code that does not access shared memory. Conversely critical sections refers to code that does access shared memory or parts of a thread that access shared memory. By considering non-critical sections as larger blocks

of sequential code that will always execute in the same order then each thread becomes comprised of blocks of non-critical code and the critical sections. This eliminates much of the complexity involved in the test scheduling problem, however testing complexity remains unchanged, only the size of the exponent, K, has been reduced.

By extending that solution further we can eliminate even more possible schedules.

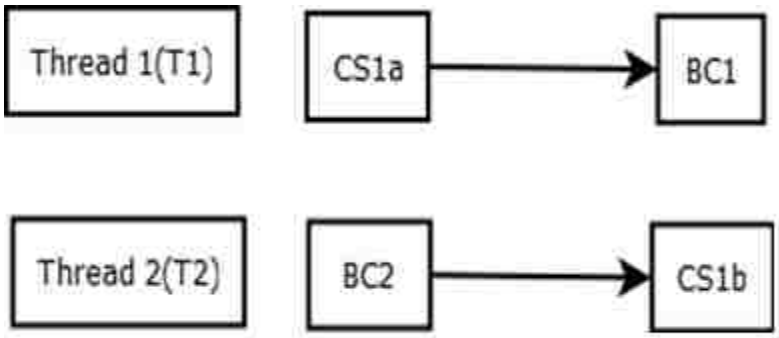


Figure 1: Simple two thread setup

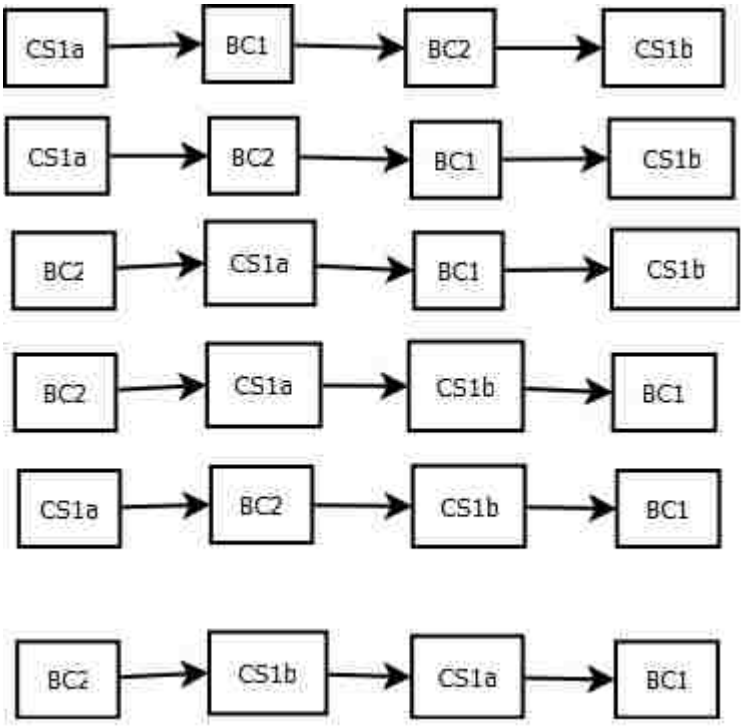


Figure 2: All possible schedules for the threads in Figure 1

Consider the threads in Figure 1. If there was a data race condition whenever CS1a is run before CS1b then the first three schedules as shown in Figure 2 will all result in the same data race condition. For that reason without a loss of generality we could say that schedules 1 through schedule 3 are members of the same equivalence class (the schedules produce the same output for identical input). We therefore need to test just one of these to sufficiently test this ordering of threads. Similarly we could say that schedule 4 and 5 are in another equivalence class. Without a lost of generality we can say that there is a possible data race condition whenever CS1a is executed before CS1b or the other way around. In this example utilizing the concept of equivalent classes reduces the 6 distinct schedules down to a total of 3 possible equivalent classes requiring just three test cases.

This concept of detecting equivalence classes to reduce the number of test cases forms the focus of the thesis. By utilizing the very nature of multi-threaded programing an algorithm will be described that can determine the equivalence classes based on similar critical section schedules for a given program so that a complete but minimal set of tests can be identified for the set of threads. The algorithm presented in this thesis is not focused on finding all data race conditions but instead is focused on presenting an algorithm that will find all equivalence class schedules.

Literature Review

In the field of parallel programming much research has been devoted to solving the basic problems of parallel code. The two main areas of research are race detection and effective testing methods. For this thesis these two areas are of great importance because the algorithm developed must utilize an effective race detection system, and it is also

important to understand what has been done previously and why the proposed algorithm is better.

First, I will focus on the different ways researchers have proposed to detect race conditions. Recall that race conditions mean any time two threads of a program access the same shared memory in ways that can result in invalid values. The major research in race detection has been conducted in ways to detect these problems at either compile time or run time.

For race detection the field can be subdivided into two main categories. The first being compile time detection of race conditions (1; 2; 3; 4). A primary example for this subfield is the paper by primary author John Mellor-Crummey entitled "Compile-time Support for Efficient Data Race Detection in shared-Memory Parallel Programs" (1). In this paper the author presents a new tool called ERASER whose goal is to improve the detection of race conditions while at the same time not adding too much overhead to program execution. The author focused his work on the FORTRAN language. To achieve these goals the author created a tool to keep a history for each shared memory variable used by multiple threads. In addition, the tool will keep track of any local variables that are passed to procedures. A slight alteration to these histories is that if multiple references inside a single statement are accessed then only a single reference needs to be maintained. With these histories, and some strategies that eliminate some of the variables that don't need to be checked, the author has created a system that appears effective for the programs tested in the paper. The key to the achieving the results found were the combination of the strategies that eliminated some of the checks that needed to be performed and thus critically reduce complexity.

Other researchers have examined different ways to detect data race conditions, from looking into static conditions, to methods utilizing locks (5; 6; 7; 8). Eli Pozniansky and Assaf Schuster looked at detecting races in an on-the-fly operation in the paper titled "Efficient On-the-Fly Data Race Detection in Multithreaded C++ Programs" (9). For this approach the authors don't look for races at compile time but only when the program is executed. The reasoning here is that it is only when the program is executed that a data race happens. The rationale behind that argument is that each time a program executes the order in which the lines of code are executed may be different, and the exact schedule needed for a data race to occur may be rare. In this paper the authors present a new tool they called MultiRace. This tool utilizes a set of optimized algorithms called Djit+ and Lockset. These algorithms are designed to detect possible race conditions. Djit+ works by logging all accesses to shared memory and looks to see if that memory has been accessed before; the authors call this a "happens-before" event. The second algorithm that used is called Lockset. This algorithm is designed to look for violations of standard locking rules. This means that each time a variable is accessed Lockset checks to see if it has been previously accessed and no unlocks have been created. This means that in the end Lockset will find all possible race events, and Djit+ will find all apparent race events. By combining these two sets a tester will have sufficient information to determine if a program can be trusted. The authors claim that MultiRace is the only tool that will return both a superset of all possible race locations, and a set of all race events.

Now we must consider how to conduct effective testing of parallel programs given the detection of race conditions. The major problem here is the nondeterministic way that a parallel program may execute, thus making standard software testing techniques

invalid. Recall that for a concurrent program a complete test is when every possible schedule has been tested.

The main problem is created by concurrent programs is the fact that each execution, test or otherwise, may result in a different schedule occurring. For that reason a lot of work has been conducted on proper ways to test these concurrent programs (10; 11; 12; 13; 14). Koushik Sen, in the paper titled "Effective Random Testing of Concurrent Programs," proposed a way to effectively test a program using random methods (15). As the author stated the only other method for testing concurrent programs is to run multiple tests in hopes that schedules are run that will produce errors. With that in mind the author developed a system that will make running random tests more efficient making errors more likely to be found. To accomplish this the author first explores different methods to limit the number of schedules that will be examined. To that end the author considered the concept of partial order reductions.

The idea of a partial order reduction is that different schedules are equivalent if the non–interacting instructions are the same. This means if two different schedules have the same ordering of non–interacting instructions then the overall order is the same. With this in mind the system developed by the author, called RAPOS, is a random partial order sampling algorithm that will sample only from a set of partial order schedules. In this manner fewer schedules must be considered which means that the likelihood of an error being found is greater. To achieve this goal RAPOS will sample any valid schedule that has probability of being selected that is not equal to 0, which should mean that RAPOS will sample the partial orders more uniformly.

Now that I have examined the work done in regards to both race detection and testing of concurrent programs I will look at why these approaches are limited and do not solve the requirement for complete testing. For software to be considered completely tested you must be able to test all possible execution paths or sequences of statements within the software at least once. The problem with concurrent programs is that to do this you must find all possible schedules. The work that has been done to detect race conditions have been limited to either compile time or run time detection. Both of these methods are valid and useful in some cases, but with regards to complete testing of a program, they are very limited. The run time detection will only find race conditions when the program runs, and the compile time detection will only finds races when the software is compiled.

In the same vein the current state of testing is very limited. The major problem with the concept proposed by Koushik Senis is that it cannot be proven that it will uniformly select a partial ordered schedule. A second major problem with this approach is that it still relies upon a random selection. This means that if only one particular schedule or group of schedules will cause an error there is a chance that it may not be selected due to the randomness of schedule selection.

Still, given the problems with the research cited above this work provides a basis for this work. Background research conducted on race detection provides me clues to an effective and reasonable race detection algorithm. Also previous research on effective random testing provides an understanding of how to process the code base to find similar schedules and how best to do the calculations.

Having covered some of the work that has been done on parallel code the focus will switch to the mathematical concept of *Linear Extensions*. *Linear Extensions* are used in

topological sorting problems (16; 17). This type of sorting problem deals with finding the linear ordering of a directed graph where the directed graph has no directed cycles. A graph that has these properties is called a directed acyclic graph.

Graham Brightwell and Peter Winkler showed in their paper entitled "Counting Linear Extensions is #P-Complete" that the problem of counting the number of linear extension is a #P-Complete problem (18). First, a #P is a different complexity class from the standard NP problems that computer science typically considers. While an NP problem looks for the answer to a question, a #P problem wants to know how many answers are there. Given that difference it is easy to understand why counting linear extensions is considered an element of this complexity class. To show that this problem is #P-Complete the authors present a lemma that simply shows that for any n >= 4 the product of primes between n and n^2 is at least $n!2^n$. Using that result they then turn their attention to counting *Linear Extensions*.

For that problem they focused on the 3-SAT Count. This problem has m variables and n clauses. For their problem they created a problem where there exists a poset P of size 7n+m. They then task their algorithm to calculate L, the number of linear extensions of P. They will then find a set S of primes that is between 7n+m and $(7n+m)^2$ that will not divide into L. To do this they first start with an Oracle O(t) that will return the number of linear extensions of a P that is at most size t. For this problem the authors set $t=(7n+m)^3$.

The authors then the set of primes for P that hold true for the lemma to be the set S. They then defined a partially ordered set Q(p) where p is an element in S. The showed that for Q(p) it can also be solved using O(t^3). They go on to show that once every step of the process is taken into account the total complexity to count all linear extensions is

O(t^14). The authors stated that with some refinement the complexity is closer to O(t^3). This shows that the problem of finding all schedules of a parallel problem is at least #P-Complete but in general it is believed to be a #P-hard problem due to complexity of the interleaving of the schedules.

The concept of *Linear Extensions* leads to the next problem of understanding topological sorting. The problem has been we studies (19; 20). The work by A. B. Kahn, in the paper entitled "Topological Sorting of Large Networks" (21), presents an algorithm to solve a topological sorting problem. His approach is to create a general algorithm that was design to be used on an IBM 7090 computer but one which is general enough that it can be implemented for a wide set of problems. This algorithm looks at solving problems that deal with so called PERT networks. A PERT network is made up of nodes that are connected by links that may or may not be directed. Let's assume we have a network with 2 nodes: A and B. If A is connect to B with a link that goes from A to B only then we can say that A is a predecessor of B and B is a successor of A. Let's add a third node called C that is connected to B with a link from B to C then we can say the activity from B-C is a successor of A. With this basic understanding the algorithm is comprised of two interlinking tables. The first table is a activity table where the activities are grouped by predecessor. The table includes a column that links it to the second table which holds the events. The activity table also includes a second column that determines if it is the last element of a grouping. The event table includes two columns, one that links it to the activity table and one that counts up the number of predecessors. The second column in the event table that links it back to the activity table will contain either the index of the first element of the group for that event or a value representing no activities for a given

event, which means that that event is not a predecessor of any other node see Appendix A for examples of these tables.

The algorithm then loops over the lists and orders them into the most desired order, as the author calls it. For the algorithm to end one of three events must have taken place. Either all events have been ordered so you now have the most desirable order or a loop was found in the graph, the final event is that a new segment must be accessed if segments where used. A segment means that the graph was divided into parts and only part at a time is considered.

These last two papers show the history behind the concept of *Linear Extensions*. The first paper shows that the general problem of *Linear Extensions* is a #P-complete problem. While the problem considered here is not as simple, it is still a member of the *Linear Extensions* group of problems. This means that at the very least the specific problem is at least #P-complete. The major difference between the standard *Linear Extensions* problem and the scheduling problem presented here is that in the present scheduling problem involves multiple independent graphs rooted at the first level.

The solution A. B. Kahn proposed is a unique solution showing that general algorithms can be used to solve #P problems. While his solution was focused on finding a general algorithm for the standard *Linear Extensions* problem that deals with a standard directed acyclic graph it does show that the concept of utilizing a general algorithm has been used in the past. It also gives some hints to the different was to represent this type of graph in a data structure.

Implementation Background

Now that background work has been considered in the area of parallel programing, the current solution can be presented. To understand this solution a deeper look at the problem is required. First, the way parallel programs work and the underlying structure of the different threads must be considered. In the field of parallel programing there are multiple types of parallel programs. This work focuses on shared memory programs parallel programs. These programs are characterized by lines of code executing simultaneously that require access to a common shared memory space.

```
Thread 1
{
        Read Temp
        Convert Temp to readable form
        Save Temp to Shared Memory
}
```

```
Thread 2
{
        Read Temp
        Convert Temp to readable form
        Save Temp to Shared Memory
}
```

Figure 3: A two thread set up that reads in a temperatures and then convert the readings before saving it to shared memory

It is that shared memory that allows the data race conditions to occur. As described earlier a data race is any time two or more threads execute in such a way that that the value in the memory may become invalid. For example, consider a storage room with two temperature probes. If a program which has two parallel threads both of which reads in

```
Read Temp (Thread 1)
Convert Temp to readable form (Thread 1)
Read Temp (Thread 2)
Convert Temp to readable form (Thread 2)
Save Temp to Shared Memory (Thread 2)
Save Temp to Shared Memory (Thread 1)
```

Figure 4: Possible schedule for the threads in Figure 3

the temperature from the probes and save the results to a single shared memory location, with the goal that only the newest reading is in the memory, pseudo code might be written as shown in Figure 3. Depending on the order in which the lines of the different threads are interleaved the value in memory may not be correct. For example if we run the first two lines of code from Thread 1 and then the system switches over to Thread 2 and runs all the lines of code before finishing the last line of Thread 1, the value in the shared memory should be the reading from Thread 2 because it is the newest reading. However due to the order of execution the final value stored in the shared memory location will be the value from Thread 1. This order is shown in figure 4.

```
Thread 1
{
        Block of NonCritical Code
        Save Temp to Shared Memory
}
```

```
Thread 2
{
        Block of NonCritical Code
        Save Temp to Shared Memory
}
```

Figure 5: Simplified representation for the threads in Figure 3

It is at this point that first clue to the solution described here was found. If we examine the code in Figure 4 we can see that the only two lines that truly affect the shared memory location and produce a race condition are “Save Temp to Shared Memory”. This

```
Block of NonCritical Code (Thread 1)
Block of NonCritical Code (Thread 2)
Save Temp to Shared Memory (Thread 2)
Save Temp to Shared Memory (Thread 1)
```

Figure 6: Possible schedule for threads in Figure 5

means that the orders in which the remaining lines of code are executed are not critical to determining if a data race condition. This concept allows us to simplify the overall problem. If we can abstract away non-critical lines then we will be left with the pseudo

code in Figure 5. This simplified code can still create the exact condition that was shown in Figure 4, the only difference is that instead of have a total of 6 lines to consider we are left with only 4 as shown in Figure 6. This concept has eliminated some of the complexity of the ordering.

Before continuing with the solution consider the way an operating system might interleave the multiple programs running at any given time. For this discussion we will be focusing on a system with a single processing core only. When a user uses a computer

Read Temp (Thread 1)
Read Temp (Thread 2)
Convert Temp to readable form (Thread 1)
Convert Temp to readable form (Thread 2)
Save Temp to Shared Memory (Thread 2)
Save Temp to Shared Memory (Thread 1)

Figure 7: Another schedule for the threads in Figure 3

they may have multiple programs executing together even though the way a system really works is that the processor can only execute one line of code at any given time. This means that the OS must be able to interleave the multiple programs together. It does this through the use of scheduling algorithms designed so that no one program takes up to much of the processing time while the other programs wait which gives the user the impression all the programs are executing at the same time. This interleaving is seamless to the user.

Block of NonCritical Code (Thread 1 and Thread 2)
Save Temp to Shared Memory (Thread 2)
Save Temp to Shared Memory (Thread 1)

Figure 8: A simplified schedule for figure 7

With this concept in mind we can now go back and look at the schedule in Figure 6. Before further examining the solution to be presented here, consider whither any functionality has been lost by blocking non-critical code. Considering the schedule in Figure 7 we can see that it is still possible to for this schedule to give up the solution in Figure 6. However, if we continue to block up non critical sections as such that we get to

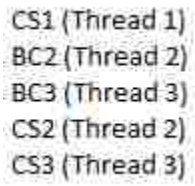


Figure 9

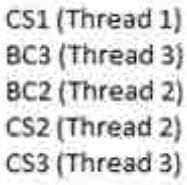


Figure 10

the schedule in Figure 8 we can being to see that we loss the ability to get all possible schedules.

Now we have blocks of blocks of non-critical code. The problem remains in that we cannot get all possible solutions if we are to assume the scheduler would execute each block as a single unit. However, a block of code comprised of individual blocks can be scheduled by the operating system scheduler in any order so long as they are run before the final two lines.

The idea of creating blocks of blocks must consider the ordering of the blocks as this is outcome determinative. Consider the ordering in Figure 9 and Figure 10; the only difference is that in Figure 10 the lines from thread 3 are executed before the lines from thread 2. If we are to assume that the underlying operating system algorithms will schedule the lines of code that makes up the blocks in any legal order then these two different schedules will be equivalent.

This idea of equivalence is key to the solution presented here. However, it brings up a dilemma: how are two classes determined to be equivalent? The answer to this lies in fact that race conditions depend only on the location of the critical sections of code. This means that for any two schedules composed of blocks of non-critical code and critical sections to be equivalent then the locations of conflicting critical sections must be the

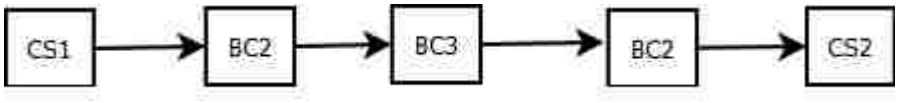


Figure 11: Partial schedule where two critical sections conflict

same. A conflicting critical section is any critical section that accesses the same shared memory as previously executed critical section.

This definition includes an additional requirement, namely the need to identify which critical sections actually conflict. This can occur when a a schedule is being built by adding a critical section that does not conflict with any of the previously added critical sections. For example in Figure 11 we can see a partial schedule where two critical sections defined as CS1 and CS2 have been added to the schedule. If CS1 and CS2 access

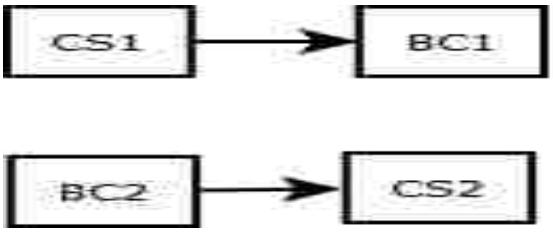


Figure 12: A graph representation of two threads

different locations in shared memory then the order in which they execute does not affect testing results. This can be extended if a conflict accrues because of a line such as “Save Temp to Shared Memory” in Figure 6. It does not matter if the “Save Temp to Shared

Memory" line from thread 2 happens directly before the "Save Temp to Shared Memory" from thread 1, just so long as it happens before it.

With this definition in mind consider how schedules can be best represented in order to best find the equivalence classes. This is important so that when the algorithms specified the type of data structure to use to represent the schedule is efficient in storage space and access time. With this in mind, recall that each thread can be represented as a

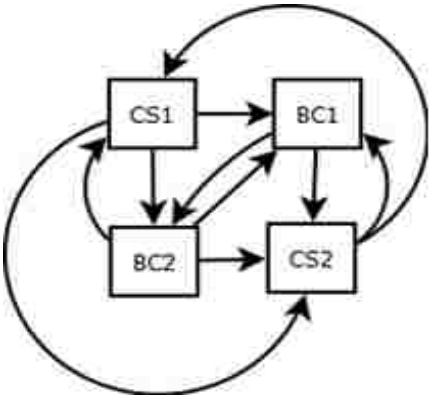


Figure 13: Complete connected graph of Figure 12

linear graph. Figure 12 shows two different threads represented as a linear graphs with

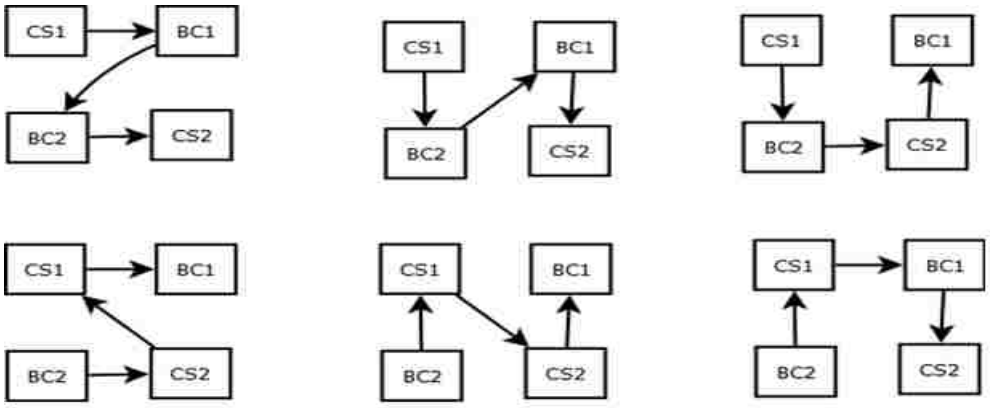


Figure 14: All possible Directed acyclic graph from Figure 13

the root being the first object in the thread.

If we then combine each of the different threads with the result is a directed graph with each graph being rooted at one of the initial elements. However, if all schedules are

represented in a single graph, the graph contains multiple loops and cycles as shown in Figure 13. For that reason it is easier to represent each schedule with its own graph as shown in Figure 14. In this manner the result would be *K* different graphs, each one representing a possible schedule.

Unfortunately, this model does not lend itself well to a data structure. There exist K different graphs to consider when running the algorithm. It is for this reason a graph representation will not work. However, this path did lead to a more plausible data

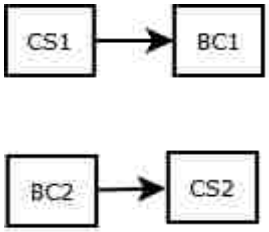


Figure 15

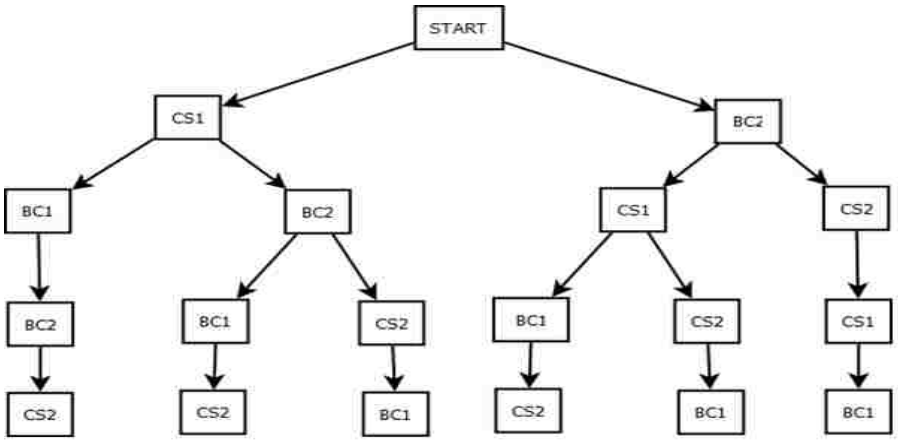


Figure 16: Tree representation of all schedules of the two threads in figure 15

structure. Considering each graph in Figure 14 it can be seen that all the ones that start with BC2 have a similar structure. Given this concept it is easy to see that combining those graphs would result in directed graph rooted at the same node. This concept led to the idea that while a graph would not work a tree structure would. In this type of data

structure at each tree node a choice is made and the branches below that node represent the choices that can be made at the node as traversal occurs. For example with two threads and at the highest level there are two choices that can be mode. If given the the threads in Figure 15, it is possible to produce the tree shown in Figure 16. This demonstrates that the number of choices is equal to the number of threads that still have code remaining to be executed.

Now that a data structure has been identified, it is time to consider the how equivalence classes will fit within the structure. If the partial tree in Figure 17 is

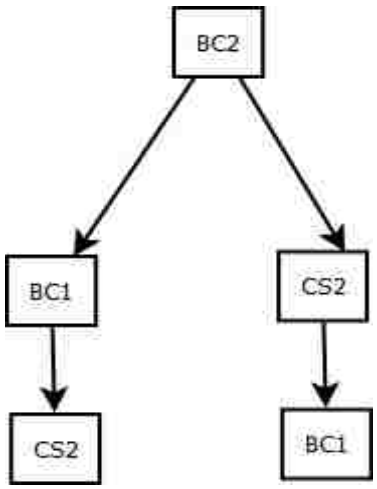


Figure 17: Single branch from figure 16

considered, it can be seen that the only difference between them is that the last two nodes in each branch are reversed. This shows that as building the tree progresses equivalent schedules can be captured by examining the different branches of the tree to see when two branches as similar and differ only in the ordering of certain blocks. In the above figure each block represents a new section of code that has been add to the tree and completed schedule will be a unique branch of the whole tree from root to leaf.

Algorithm Design

With the data structure selected and access defined, the entire algorithm can be described. First a key concept in algorithm design is functional decomposition. In this case the algorithm needs to be designed in two parts. The first part will be responsible for parsing the parallel program into the proper format while the second part will be responsible for finding all of the schedules for a given program.

The first thing to consider in building these two algorithms is time. While, the first algorithm is relatively simple the second one has significant complexity. With the concept of utilizing a tree structure for finding the schedules the complexity of building the tree is at least O(n log n). With this in mind decisions must be made as to the building the tree and when to search for the equivalence classes.

A logical solution is to build all the trees for a given program and then parse the resulting schedules to find the equivalent classes. However, while this plan is simple, the potentially huge number of possible schedules makes it extremely difficult and time consuming to perform. A better approach is to prune the tree as it is built so that unneeded branches are not traversed. The algorithm requires detection of a few pruning points and implementation of pruning.

Considering just the adding critical sections, it can be seen that when a critical section is added that conflicts with a previous critical section then the algorithm must ensure that an equivalent path has not already been found. Secondly the algorithm must detect when an equivalent schedule has been created. Returning to the tree in Figure 17 we can see that the two branches are equivalent if CS2 does not conflict with CS1. This means once the second branch is created we must compare it to previous branches so we don't create any redundancies in the final listing.

Now that detecting when tree pruning should occur due to the equivalent schedules, consideration of when to prune illegal trees must be done. Given the overall program flow in Figure 18 it can be seen that the starts with a single main thread that then spawns a group of 4 threads which does some work and then comes back together in the main thread, which does some more work before spawning a second group threads. It is not possible for a thread from the second group to be executed before the original threads have complete work. This means that detection of any branch where a block of code from a thread cannot possibly be executed allows elimination of the remaining branches due to the fact that they can't be possible occur in a legal execution of the program. The next two sections consider the two main parts of the algorithm.

Parsing Algorithm Design

The parsing algorithm is the part of the algorithm where the data organization takes place. Code is never in the format that the schedule finding algorithm requires. For that reason a parsing algorithm is needed to turn normal code into the format needed by the scheduling algorithm. To do this the algorithm accepts as input the files containing program code. As output it delivers to the scheduling algorithm an array of thread objects where each object is comprised of blocks of non-critical code and blocks of critical code. As before critical code is any part of the source code that accesses shared memory.

In addition, each block of critical and non-critical code includes state based information. This state information is divided between updating the state program and a minima state required. For example a block of non-critical code may contain a section that deals with synchronization between multiple threads allowing the code to update the state of the program. The other a block of code may only be executed after the program

reaches a given state. This information helps to carry over the time component of a program. The time aspect of a program deals with the fact that not all threads are created at the same time. Some threads may be created earlier or later in the program then other. The staggered creation concept is what I mean when talking about a the time aspect of a program.

The parsing algorithm is very simple. By default, a global section of code is added to the array in index position 0. The section is used to define the fact that every program contains some information in the global section of the program. The next step is loop over each line of code and checks to see what type of line is it. In general there are two types of lines, lines that are part of thread and lines that are part of a global section of code. If a given line is the start of a thread then a new thread object is added to the array and we loop over that thread until we have reached its end. If a line is part of the global section of code then it is added to the global section.

For simplicity the global section is considered a thread during the execution of the scheduling algorithm. The only difference in parsing threads and the global section is that a global section may include some synchronization lines while threads do not. Other wise the process is the same for both types.

First the algorithm checks to see if the current line is a critical section. If it is a critical section then it is added to the current thread . If the thread had previous non critical section block of code then that block of code is closed and a new critical section block is added. If there was not a previous block then a new critical section is added. On the other hand if the current line is a non-critical section then the opposite happens. If there is a

currently open block of non-critical code then it is added to that, else a new block is created and it is added to that block.

In either case the current state of the program is passed to the thread object when a new block is added. This state information is included with the block of code. A block of code contains two pieces of information. If a section of the block contains a synchronization point then it also contains the new state of the program. In addition it also contains the state of the program before execution. In this manner it can be determined when it is legal to execute a given line of code. If it is a block of non critical code it contains the earliest allowed state and the most recent state. For example if a block of code contains three lines of code and each line updates the state then the block will contain the last update. If that same example has three different current states then only the earliest will be recorded. For a complete listing of this code see Appendix B.

Schedule Finding Algorithm Design

The schedule algorithm contains the critical part of the solution where most of the work is done. Recall that the parsing algorithm outputs a set T which is a set of all the parsed threads of the program. With that set it is easy to determine the total number of threads in a program. The total is incremented by 1 because we need a thread that deals with the non-threaded part of the code.

First, the algorithm loops through every initial block of each thread in the set T. For each element the algorithm determines if an update to the current state of the program is needed. This state is the variable that governs if a block of code may be executed. If the state needs to be updated then that state variable is changed. Once this part of the

algorithm completes the next step is to determine if the current block is a legal block to execute at this point.

Once an initial block of code that is legal to execute is found construction of a new schedule begins. The first step is to create a new empty schedule, then determine if the block contains conflicting critical sections or a non-critical-section. If it is a non-critical section or a non- conflicting critical section it is added to the general code block. The final step in this initial algorithm is to call the recursive function to build the tree. To do this the algorithm loops over each thread and calls the recursive function with the index being passed in. The recursive function takes in the set of threads T, the number of threads, an array to hold the final schedules found, the current schedule, the index of the next block to add, the state variable and array of current spot in each thread. See Appendix C for the complete code for this function.

The next function called is the recursive function responsible for building each schedule. This function does most of the work in the algorithm. The function takes in the values from the previous function and then recursively builds a schedule. The first step is to index for the current thread. It then checks to see if the next block is a valid block to add to the schedule at this time. If it is not valid it will exit this recursive call. If it is valid it will update the state variable appropriately.

Once the state value has been updated then the algorithm checks to see if the new block conflicts with any of the previous blocks that have been added to the tree. If it does conflict then it will add the block as a conflicting section. Once added the algorithm checks the final schedules that have already been found. If a previous schedule equivalent schedule exists then this branch is terminated.

If the algorithm is not done then the next step is to make sure we are not at the end of a schedule. This is done by checking that one thread remains with more blocks to add to the schedule. If more blocks remain the function is called again just as was done in the previous algorithm.

If no more blocks remain to add to the schedule then just two final remain. First, one final pruning takes place to ensure the current schedule is not equivalent to any others and if it is not then it is added to the list of final schedules. For a complete listing of this function see Appendix D.

Large Scale Example

An example will illustrate the algorithm in detail. For this walkthrough the threads in

T0: BC0, CS0, BC1, BC2
T1: BC3, CS11
T2: BC4, CS12
T3: CS23
T4: CS24
T5: CS25

Figure 18: Six threaded example

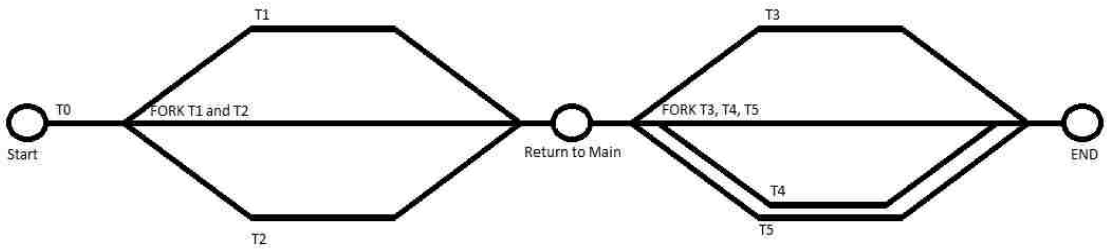


Figure 19: Time component of for threads in Figure 18

Figure 18 will be used. As shown in figure 18, 6 threads comprised of a total of 11 blocks

of code comprise the program. The time schedule for this example is given in Figure 19. As we can see we have a main thread, T0, which spawns 2 threads T1 and T2. After those threads finish the T0 spawns three more threads which then finish before the main thread ends. In addition, blocks BC0, BC1 and BC2 both change the state of the program. BC0 changes the state from start to S1 and CS0 changes it to S2 and BC1 changes the state to S3. CS11 will change the state to either S1ab or S1a depending on the initial state (either S1b or S1). CS12 will change the state to either S1ab or S1b depending if the state was S1a or S1. BC3 and BC4 can happen after the state is S1. CS0 is limited to happening after the state is S1ab. BC1 is limited to happening after the state is S2. Let's also assume that CS23, CS24, CS25 can only happen after the state is S3.

With the example defined the algorithm can be executed. First the algorithm will loop over each of the 6 threads. For each thread it will check to see if the first block in the thread will update the state of the program. For BC0 it can be seen that the state is updated from start to S1. Next, it will check to make sure that it is legal to start a schedule with BC0. We can see that it is legal for BC0 to start a schedule. With that information it will create a new array that will contain the current index for each thread. For BC0 from thread 0 the array will contain [1,0,0,0,0,0]. If there was only one block in thread 0 then the array will be [-1,0,0,0,0,0]. A "-1" in any spot represents the condition where the thread is out of blocks of code.

So far for BC0 the algorithm has update the state, and it created a new indexing array. The next step is save BC0 to the new schedule. The first thing the algorithm will do is to determine if the block is a critical section or a non-critical section. Since BC0 is a non-critical section it will just add the block to the schedule without any indication. next, the

algorithm will loop over all the threads again and pass the new schedule, the current state and the index of the next thread to the function that will build the schedule.

Now that the algorithm has handled BC0, let's consider how it will handle the rest of the initial blocks. BC4 is the first block of thread 1 and can only happen when the state is in position S1. However, given the algorithm has started a new loop the state has been reset to start. This means that BC3 is not a legal block to start a schedule. The same thing will happen with the rest of the 4 threads as well due to the fact that each of them must accrue after a state that has not yet happened. Now that it is has been observed that the first function will result in only a single root node it is possible to see the importance of having the time based pruning function as part of the main algorithm. As seen in Figure

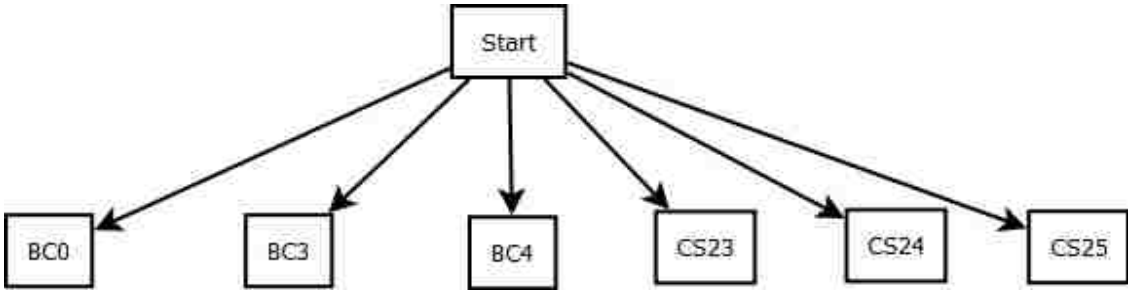


Figure 20: Starting tree structure for the threads in figure 18

20 it can be seen that already 80% of the possible schedules have been eliminated because they are not legal.

The next step is to return to the buildTree function call with BC0 as the start of a schedule. The next step is to verify that the next block to be added to the schedule is legal. Recall the current schedule is passed to this function along with the thread index of the next block to be added to the schedule, and controlling array of the location in each thread considered so far. This function builds the sub-trees under BC0 as shown in Figure 21. The first branch to test will be to add CS0 to the schedule. This is not a legal move

because that state is still only S1. This means we will skip adding that block for now. The next block that will be tested will be to add BC3. When we test this block we can see that it is a legal move.

The algorithm will then check to see if BC3 will update the state, which it does not. The next step is to see if BC3 is a critical section which it is not so BC3 will be added to the schedule. Because the tree was not pruned the algorithm will continue to add more blocks to the schedule. The first thing it will do is update the controlling index array. The algorithm then checks to make sure there exist more blocks to add. It does this by making sure that within the index array there are values that are not -1. Since currently the index array is [1,1,0,0,0,0] more blocks remain to be added. The algorithm then loops over all the threads again and tries to add more blocks to the schedule.

Pausing to review Figure 21, it can be seen that at the same level where BC3 was added to the schedule BC4, CS23, CS24 and CS25 should also be considered for addition. Only BC4 is a legal move but before adding this block the entire BC3 branch must be considered.

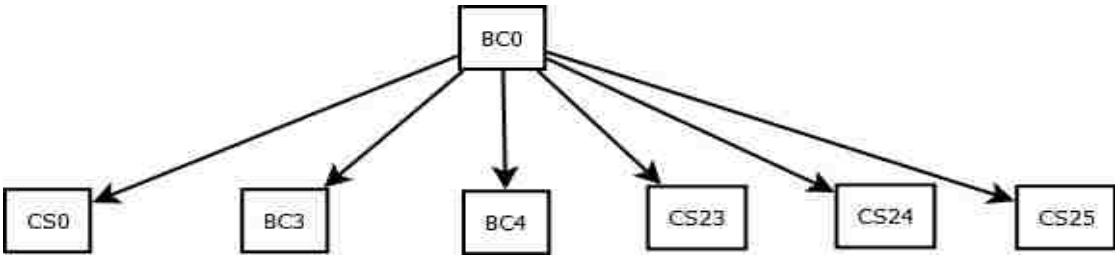


Figure 21: Secondly of tree down a single branch for Figure 18 threads

The next step in the BC3 branch will be to add CS11. When this block is added it can be seen that it does change the state from S1 to S1a. Continuing down this branch adds CS12. This is the first critical section that conflicts with a past section. Because only two

critical sections exist there is nothing to prune. Traveling all the way down that branch provides a partial tree as shown in Figure 22 on next page.

Returning to the BC3 step and checking the second branch the algorithm will start with BC4. BC4 is added to the schedule and because it is the first part the algorithm

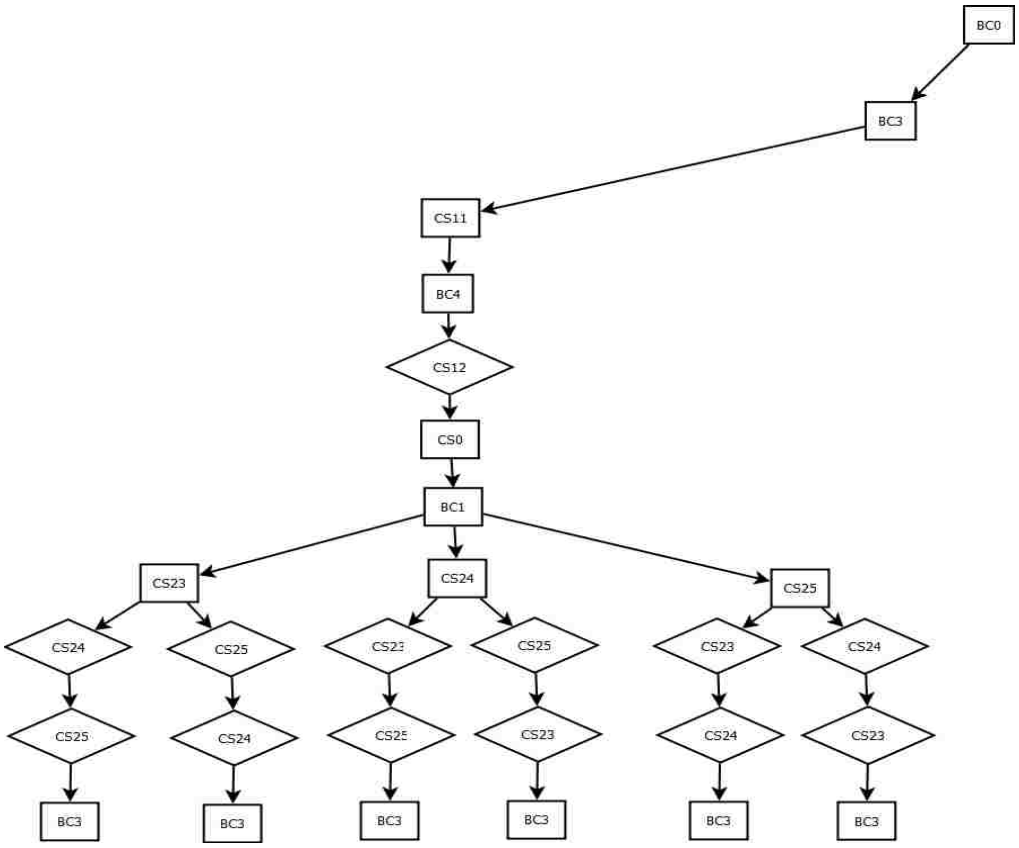


Figure 22: A completed branch from start to leaf node for Figure 18 threads

moves on to adding either CS11 or CS12 to the schedule. They are added in the order shown in Figure 23 to eliminate the first branch. When adding CS12 to the schedule the algorithm will determine if pruning is needed. CS12 is the only block that conflicts with any previously added blocks and it is in the same spot. In addition the previous blocks before CS12 are just a permutation of the previously finished schedules. This means the branch from CS12 on down can be eliminated.

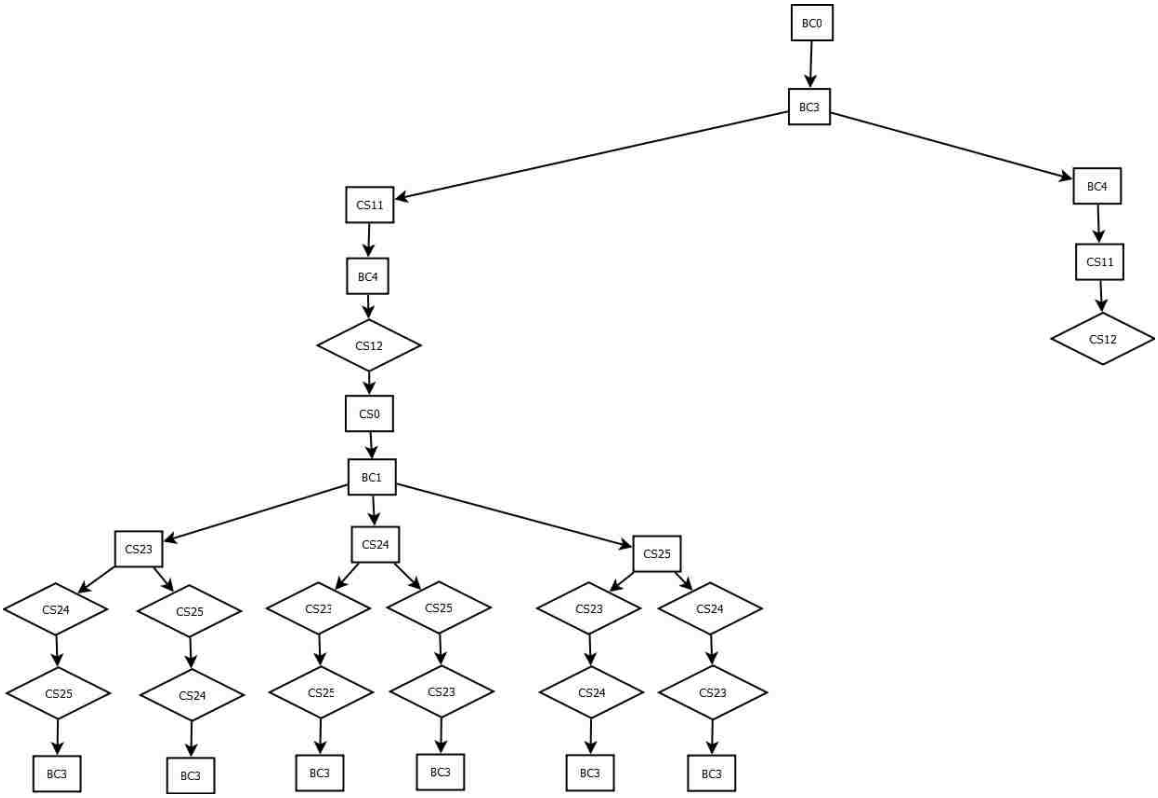


Figure 23: First equivalent branch for Figure 18 threads

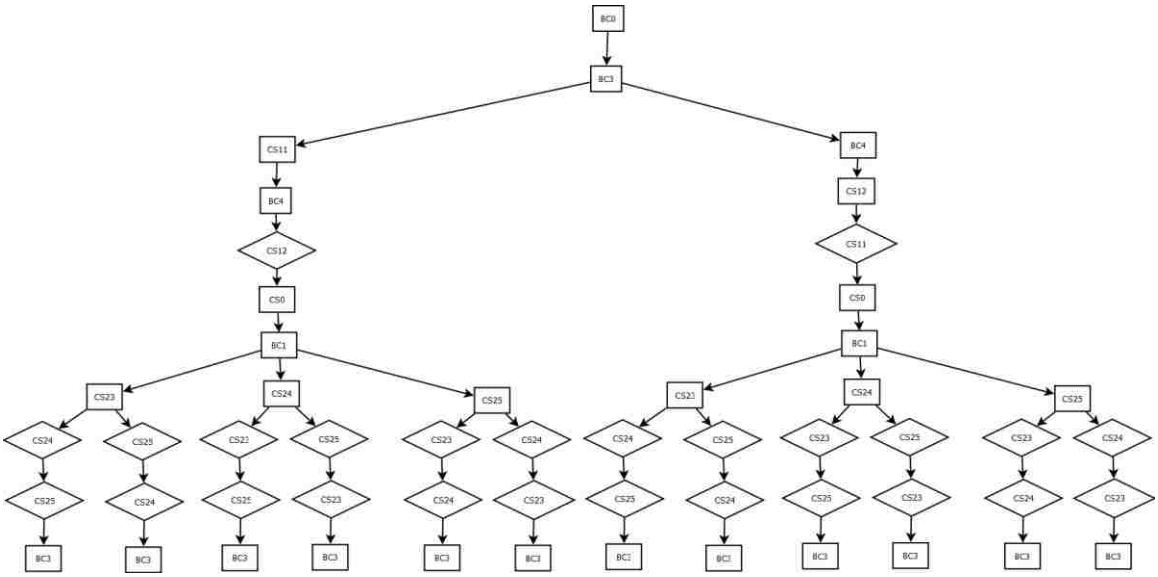


Figure 24: Two completed branches for the threads in Figure 18

Considering Figure 24 the complete tree for all legal branches below BC3 is shown.

This indicates that there are still 12 possible schedules that must be tested for this partial

tree. However that tree was built under the assumption that threads T3, T4, T5 are considered unique. Those three threads are comprised of a single block of a critical section, otherwise all three are identical. This means that no matter how they are executed T3, T4, T5 will always be run together. With that condition we are left with the tree shown in Figure 25. As we can see in Figure 25 the 12 possible schedules have been reduced to two. The only thing remaining is to make sure that when the code is parsed that identical threads are marked so that it is easy to recognize them. (The image in Figure 25 has been turned on its side for space reasons only)

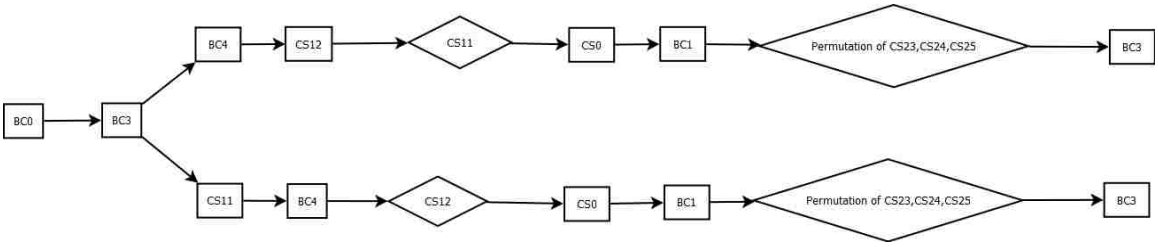


Figure 25: Possible simplification of Figure 24

Experiments Set Up

Now that I have covered how my algorithm works lets look at the setup of the experiments that I conducted to test the algorithm design. While it would be desirable to conduct complexity analysis on this algorithm the complexity of the design means that determining the algorithm complexity is extremely difficult, it is for that reason that another approach had to be utilized. The approach that I settled-on was to create a series of synthetic data sets that represents real world problems. To simplify the problem, so that a simple script could be used, the time part of the algorithm was not taken into account for most examples. The simplification was used due to the fact that the goal of this test was not to show that the whole algorithm worked but to prove that the concept works.

To determine the results of the algorithms a simple Perl script was created that could determine all the legal schedules, to make the end of a legal schedule it would place a #

```
CS1a, BC1, CS2a,
BC2, CS2b, BC3, CS1a,
```

Figure 26: Example of thread setup for the experiments

to make parsing them by hand easy. The script would read in a text file where each line represents a thread in the format that the algorithm would expect an example of this format is listed in figure 26.

```
sub build_Tree
{
	my ($tree, $currentCS, $threadNum, @currentIndex) = @_;

	#get the current index value for the thread of the block being added
	my $lineNum = $currentIndex[$threadNum];
	$tree = add_Node($tree, $currentCS, $threadNum, $lineNum);
	#open file of threads
	open(FILE, $file);
	my @raw_data = <FILE>;
	close(FILE);
	#determine total number of threads
	my $threadNumber = @raw_data;

	#get the current thread
	my $thread = $raw_data[$threadNum];
	chomp $thread;
	my @threadP = split(',', $thread);
	my $length = @threadP;
	#get current thread length amd determine if it has more blocks to add
	if(($currentIndex[$threadNum]+1) == $length)
	{
		$currentIndex[$threadNum] = -1;
	}
	else
	{
		$currentIndex[$threadNum] = $currentIndex[$threadNum]+1;
	}
	my $final = 0;
	#try and add more blocks to schedule, if you can change final value so it does not save thread
	for(my $i = 0; $i<$threadNumber; $i++)
	{
		if($currentIndex[$i] != -1)
		{
			$final = 1;
			build_Tree($tree, $currentCS, $i, @currentIndex);
		}
	}
	#if no more blocks can be added print final schedule
	if($final == 0)
	{
		print "$tree#\n";
	}
}
```

Figure 27: build_Tree function for the Perl script used

Once the script reads in the text file it then creates all legal schedules. To do this it starts the tree with the first node in each thread. It then calls a recursive function in which the next block of the tree is added, this function is listed in figure 27. Each time all possible blocks are added to the schedule one at a time, in this manner the all possible

combinations are created. For all my experiments I defined that two sections are in conflict by the following, CS#y conflicts with CS#x, where # is the same number in each example and X and Y are two different letters used to distinguish the thread that it came from.

Once all the schedules have been created I manually went through and parsed the list keeping all first member of an equivalent class. For example take the schedule in Figure 28. The first thing to do was to find the first conflicting critical section. Then I would determine all possible permutations of the blocks before that element. For the example in Figure 28 the first conflicting block is CS1b which conflicts with CS1a, the resulting

```
Threads:
BC1, CS1a, BC2, CS2a,
CS1b, BC3, CS3b,
BC4, CS4c,
CS5d, BC5, CS1d

Schedule:
BC1, BC4, CS1a, CS5d, BC5, CS4c, BC2, CS1b, CS2a, BC3, CS3b, CS1d,#
```

Figure28: Sample threads and a sample output from Perl script

```
BC1,CS1a,BC2,BC4,CS4c,CS5d,BC5
BC1,CS1a,BC2,BC4,CS5d,CS4c,BC5
BC1,CS1a,BC2,BC4,CS5d,BC5,CS4c
BC1,CS1a,BC2,CS5d,BC4,CS4c,BC5
BC1,CS1a,BC2,CS5d,BC4,BC5,CS4c
BC1,CS1a,BC2,CS5d,BC5,BC4,CS4c
BC1,CS1a,BC4,BC2,CS4c,CS5d,BC5
BC1,CS1a,BC4,BC2,CS5d,CS4c,BC5
BC1,CS1a,BC4,BC2,CS5d,BC5,CS4c
BC1,CS1a,BC4,CS4c,BC2,CS5d,BC5
BC1,CS1a,BC4,CS4c,CS5d,BC2,BC5
BC1,CS1a,BC4,CS4c,CS5d,BC5,BC2
BC1,CS1a,BC4,CS5d,BC2,CS4c,BC5
BC1,CS1a,BC4,CS5d,BC2,BC5,CS4c
BC1,CS1a,BC4,CS5d,CS4c,BC2,BC5
BC1,CS1a,BC4,CS5d,CS4c,BC5,BC2
BC1,CS1a,BC4,CS5d,BC5,BC2,CS4c
```

Figure 29: Permutations of the blocks between conflicting critical sections

permutations are listed in figure 29.

I then went through and determined the next conflicting critical section in the schedule and then found all possible permutation between the two conflicting sections. I repeated this step until I had found all conflicting sections. For this example the only conflicting sections are CS1a, CS1b and CS1d. I then found the permutations between them (Figure 30). After that I then found the schedules that had every possible combination of the

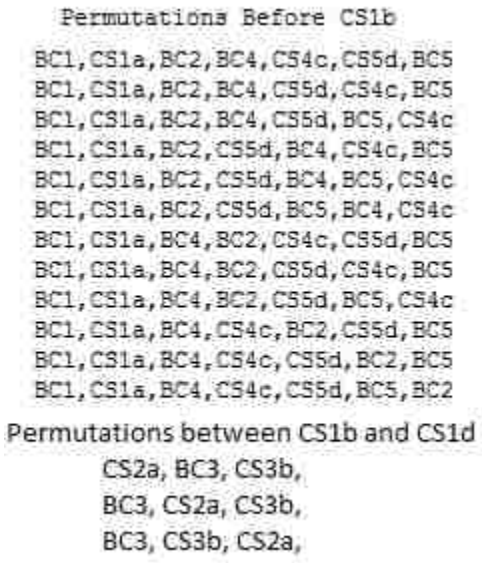


Figure 30: Permutations between next groups of critical sections

permutations and the conflicting critical sections, see Appendix E for a complete listing. The resulting schedules where then removed from the overall list, for the results in Figure 30 a total of 630 schedules where removed from the list of schedules.

Experiments

For this study I ran a total of 6 different cases. Each study was designed to try and replicate real world situations. Case 1 and Case 2 are both 2 thread systems with 2 blocks of code in each thread. In the Case 1 each thread has a non-critical section and a

conflicting critical sections, the difference between the threads is the order in which the critical and non-critical sections occur. In Case 2 it is a similar set up but each thread is comprised of a critical section followed by a non-critical section. The reason for these two cases is that it will give me a comparison between the improvement between conflicting threads and what happens when the order is changed. Case 3 is a system of two threads with a total of 5 blocks in each thread. The first thread has 2 critical sections and 3 non critical sections; the second thread has 3 critical sections and 2 non critical sections. For Case 3 a total 4 of the critical sections conflict with each other, in sets of 2. This system was designed to represent a more complex program that would access more shared memory and in complex ordering. Case 4 was the first test comprising of more than 2 threads. It contains 3 threads and a total of 8 blocks of code. It total there are 2 groups of critical sections that conflict. This case was designed to represent an even more complex system. The final case that I conducted was Case 5. It was comprised of a total of 4 identical threads. This was meant to represent a system that forks off a bench of identical threads that would do some processing and then access a shared memory space.

In addition to the 5 cases that I ran through from start to finish I also ran a series of additions cases to determine the total number of possible schedules. The reason that I did not parse these examples is due to the number of schedules. For example, one case I ran was similar to Case 5 above but it had a single extra thread. For Case 5 it found a total of 2520 schedules while the addition run that I conducted it found a total of 113400 total schedules. Due to the fact that I was conducting the parsing by hand it was not feasible to parse that examples down to only the equivalent schedules. However the runs did provide information regarding the total growth rate of the schedules.

Case 1

For the first case I examined I used a simple set up. This case was also the one I used to design and test my algorithm. This first case was comprised of the threads as shown in Figure 31. As we can see it is simply comprised of two threads where the order of the conflicting and non-conflicting sections where reversed. In this example the two conflicting sections conflict.

For this first test the amount of improvement was limited due to the fact that the number of total schedules was only 6 as shown in Figure 32. For this example I was only

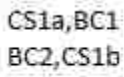


Figure 31: Threads for Case 1

able to find a total of 3 equivalent classes, as shown in Figure 33. This means that we only got an improvement of 50% when implementing my algorithm. Even though the amount of improvement was only 50% we can see that the first equivalent class replaces a total of three schedules. Similarly we can see that the next two classes replace 2 and 1 schedules respectfully.

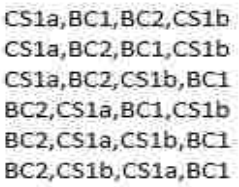


Figure 32: Resulting schedules for Case 1

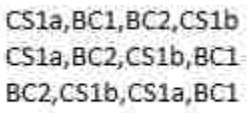


Figure 33: Final equivalent schedules for Case 1

Case 2

For the second case I examined I used a similar set up to case 1 but this time the two threads are configured similar. This case was comprised of the threads as shown in Figure 34. As we can see it is simply comprised of two threads where each thread is comprised of a critical section followed by a non-critical section. In this example the two conflicting sections conflict.

For this test the amount of improvement was limited due to the fact that the number of

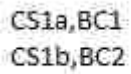


Figure 34: Threads for Case 2

total schedules was only 6 as shown in Figure 35. I was only able to find a total of 4 equivalent classes, as shown in Figure 36. This means that we only got an improvement of 33.33% when implementing my algorithm. Even though the amount of improvement was only 33.33% we can see that the ordering of the blocks of code is important when we

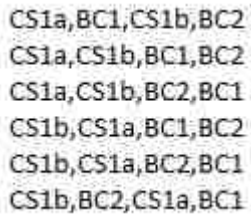


Figure 35: Resulting schedules for Case 2

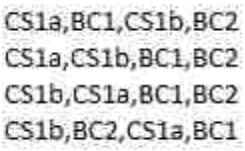


Figure 36: Final equivalent schedules for Case 2

compare this result to the results in case 1. The only difference between these two cases is the ordering of the blocks in thread 2. By changing the ordering we decreased the improvement by 1/6 and added an additional equivalent class to the final result.

Case 3

As I stated above the third case was designed to represent a more complex example then the first two cases. For that reason in the experiment I upped the total number of blocks being used and increased the complexity of the program so that more schedules would be created. The threads that were used in this example are listed in Figure 37. As we can see it is once again comprised of 2 threads but this time there are multiple critical sections in each thread and the total number of conflicting sections was increased as well.

These changes resulted in a total of 252 legal schedules a sampling of these schedules

```
BC1,CS2a,BC2,CS3a,BC3
BC4,CS1b,CS2b,CS3b,BC5
```

Figure 37: Threads for Case 3

can be seen in Figure 38, full a complete list see Appendix F. Out of all possible schedules there were a total of 22 equivalent classes. This means that my algorithm

```
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC1,BC4,CS2a,BC2,CS3a,BC3,CS1b,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,BC3,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,BC3,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC3,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS3a,BC3,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,BC3,CS3b,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC3,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3a,BC3,CS3b,BC5
```

Figure 38: Partial list of resulting schedules for Case 3

removed a total of 230 schedules which represents 91.27% of all the legal schedules. The 22 equivalent classes that where found can be seen in Figure 39 as seen on next page. As we can see in these examples that the only conditions that matter are the order in which

CS2a, CS3a, CS2b and CS3b. In no case does the placement of CS1b matter for the simple fact that it does not conflict with any other section.

```
BC1,CS2a,BC2,CS3a,BC3,BC4,CS1b,CS2b,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3a,CS3b,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3a,BC3,CS3b,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3a,CS3b,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3b,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3b,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC2,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC2,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS3b,BC5,CS2a,BC2,CS3a,BC3
```

Figure 39: Partial list of final equivalent schedules for Case 3

Case 4

For this case I wanted to expand the number of threads while still keeping the number

```
BC1,CS2a
BC2,CS3b,BC3
CS2d,CS3d,BC5
```

Figure 40: Threads for Case 4

of blocks low. It was with that concept in mind that I developed the threads that are shown in Figure 40. As we can see this case is made up of a total of 3 threads. Thread 1 contains a single block of non-critical code and a block of conflicting critical code.

Thread 2 is comprised on 2 blocks of non-critical code that surround a block of critical code. The final thread has two blocks of critical code followed by a block on non-critical code.

Once I ran the experiment I discovered a total of 560 legal schedules which can be seen in Figure 41 and a complete list in Appendix G. Out of all of those schedules I was able to bring it down to a total of 60 equivalent schedules. This provided for an improvement of 89.29%. While this test did not do as well as the test in case 3, it still provided nearly the same level of improvement even considering the fact that it utilized an additional thread. While case 3 used a total of 10 blocks of code, case 4 only used 8 blocks of code but those 8 blocks resulted in over twice as many legal schedules and 3 times as many equivalent classes of schedules. A sample of the results can be seen in Figure 42 and a complete listing is in Appendix H.

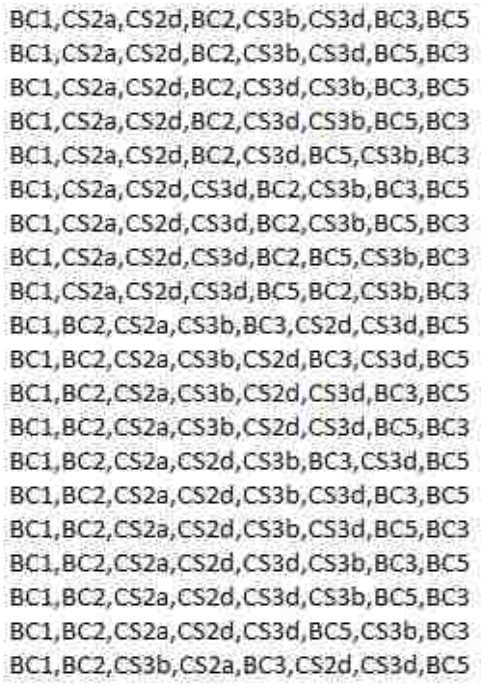

BC1,CS2a,CS2d,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3b,CS3d,BC5,BC3
BC1,CS2a,CS2d,BC2,CS3d,CS3b,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3d,CS3b,BC5,BC3
BC1,CS2a,CS2d,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2a,CS2d,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2a,CS2d,CS3d,BC2,CS3b,BC5,BC3
BC1,CS2a,CS2d,CS3d,BC2,BC5,CS3b,BC3
BC1,CS2a,CS2d,CS3d,BC5,BC2,CS3b,BC3
BC1,BC2,CS2a,CS3b,BC3,CS2d,CS3d,BC5
BC1,BC2,CS2a,CS3b,CS2d,BC3,CS3d,BC5
BC1,BC2,CS2a,CS3b,CS2d,CS3d,BC3,BC5
BC1,BC2,CS2a,CS3b,CS2d,CS3d,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3b,BC3,CS3d,BC5
BC1,BC2,CS2a,CS2d,CS3b,CS3d,BC3,BC5
BC1,BC2,CS2a,CS2d,CS3b,CS3d,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3d,CS3b,BC3,BC5
BC1,BC2,CS2a,CS2d,CS3d,CS3b,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3d,BC5,CS3b,BC3
BC1,BC2,CS3b,CS2a,BC3,CS2d,CS3d,BC5

Figure 41: Partial list of resulting schedules for Case 4

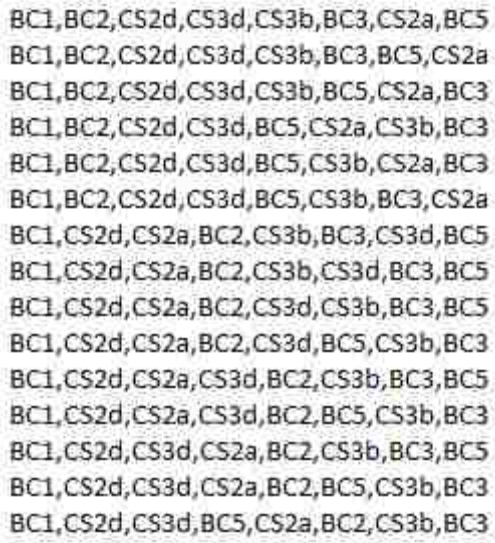

BC1,BC2,CS2d,CS3d,CS3b,BC3,CS2a,BC5
BC1,BC2,CS2d,CS3d,CS3b,BC3,BC5,CS2a
BC1,BC2,CS2d,CS3d,CS3b,BC5,CS2a,BC3
BC1,BC2,CS2d,CS3d,BC5,CS2a,CS3b,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,CS2a,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,BC3,CS2a
BC1,CS2d,CS2a,BC2,CS3b,BC3,CS3d,BC5
BC1,CS2d,CS2a,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2d,CS2a,BC2,CS3d,CS3b,BC3,BC5
BC1,CS2d,CS2a,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2d,CS2a,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2d,CS2a,CS3d,BC2,BC5,CS3b,BC3
BC1,CS2d,CS3d,CS2a,BC2,CS3b,BC3,BC5
BC1,CS2d,CS3d,CS2a,BC2,BC5,CS3b,BC3
BC1,CS2d,CS3d,BC5,CS2a,BC2,CS3b,BC3

Figure 42: Partial list of final equivalent schedules for Case 4

Case 5

The final case that I ran through all the way was the largest of the 5. It was comprised of a total of 4 threads with a total of 8 blocks but 4 out of these 8 blocks all conflict. As shown in Figure 43. Each thread is comprised of a non-critical section followed by a critical section. We can also determine that each of the critical sections access the same memory location for the simple fact that each one has the same numeric value. This case was designed to represent a system when a number of identical threads are forked off and each thread does the same thing before updating some shared memory.

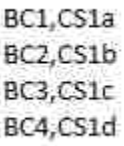


Figure 43: Threads for Case 5

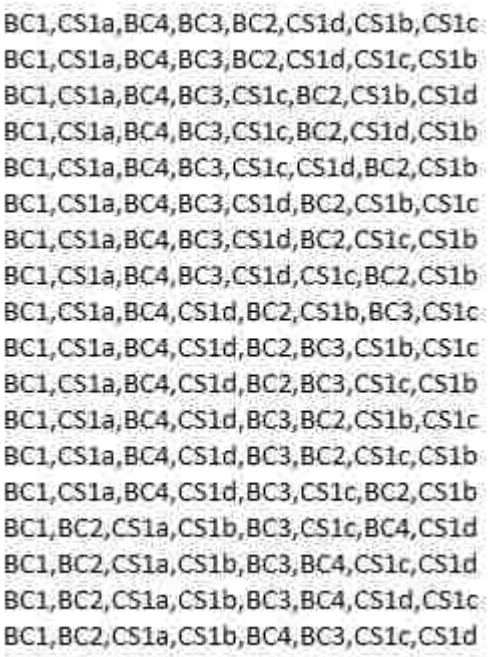


Figure 44: Partial list of resulting schedules for Case 5

Once again the total number of legal schedules increased from the previous example. It went from 560 in case 4 to a total of 2520 schedules in case 5, see Figure 44 for a partial listing. Out of those 2520 schedules a total of 143 equivalent schedules where found. This resulted in an improvement of 94.33%. As we can see from these values the

```
BC2,BC3,CS1c,CS1b,BC4,CS1d,BC1,CS1a
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1a,CS1d
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1d,CS1a
BC2,BC3,CS1c,BC4,CS1b,CS1d,BC1,CS1a
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1a,CS1b
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1b,CS1a
BC2,BC3,CS1c,BC4,CS1d,CS1b,BC1,CS1a
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1a,CS1c
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1c,CS1a
BC2,BC3,BC4,CS1d,CS1b,CS1c,BC1,CS1a
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1a,CS1b
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1b,CS1a
BC2,BC3,BC4,CS1d,CS1c,CS1b,BC1,CS1a
BC2,BC4,CS1d,CS1b,BC1,CS1a,BC3,CS1c
```

Figure 45: Partial list of final equivalent schedules for Case 5

number of schedules is not as much related to the number of blocks but the total number of schedules. For a partial listing of the equivalent schedules see Figure 45, for a complete listing of all legal schedules and all equivalent schedule classes see Appendix I and appendix J.

Case 6

Case 6 was designed to show what effect utilizing the information about the state of the program would have on the overall improvements. For this case I utilized 3 threads as shown in Figure 46. Thread 1 represents the global section of the program. The state

```
BC1,CS2a,BC2,CS1d
BC3,CS1b
BC4,CS1c
```

Figure 46: Threads for Case 6

information for this program can be seen in Figure 47. As shown the program starts and blocks BC1, CS1a and BC2 are run before threads 2 and thread 3 are forked off. After those threads finish then CS1d is executed.

Without the time based information included these threads would create a total of 420 legal schedules as listed in Appendix K. However once the time based rules are taken into account we can eliminate all but 6 schedules as shown in Appendix L. Furthermore, once

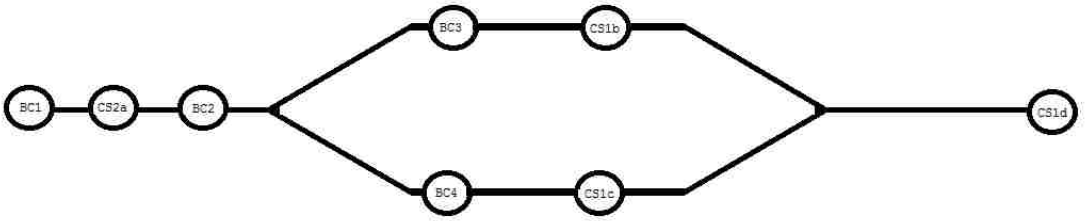


Figure 47: Time aspect for Case 6 program

we apply the equivalent class concept we can get it down to a total of only 2 legal schedules. The two equivalent classes can be seen in Figure 48.

```
BC1,CS2a,BC2,BC3,CS1b,BC4,CS1c,CS1d
BC1,CS2a,BC2,BC4,BC3,CS1c,CS1b,CS1d
```

Figure 48: Equivalent schedules for Case 6

Results

Let us now look at the collection of data that was gathered from these experiments. In every case a substantial amount of improvement was seen when applying my algorithm to the legal schedules. The total number of equivalent schedules found can be seen in Figure 49 on next page and the percentage improvements are shown in Figure 50 on the page 46. As we can see the smallest amount of improvement was in case 2 while the largest was seen in case 5.

In case 6 where the state of the program was taken in to account it can be seen that we went from a possible 420 schedules down to only 6 legal schedules. When I then applied

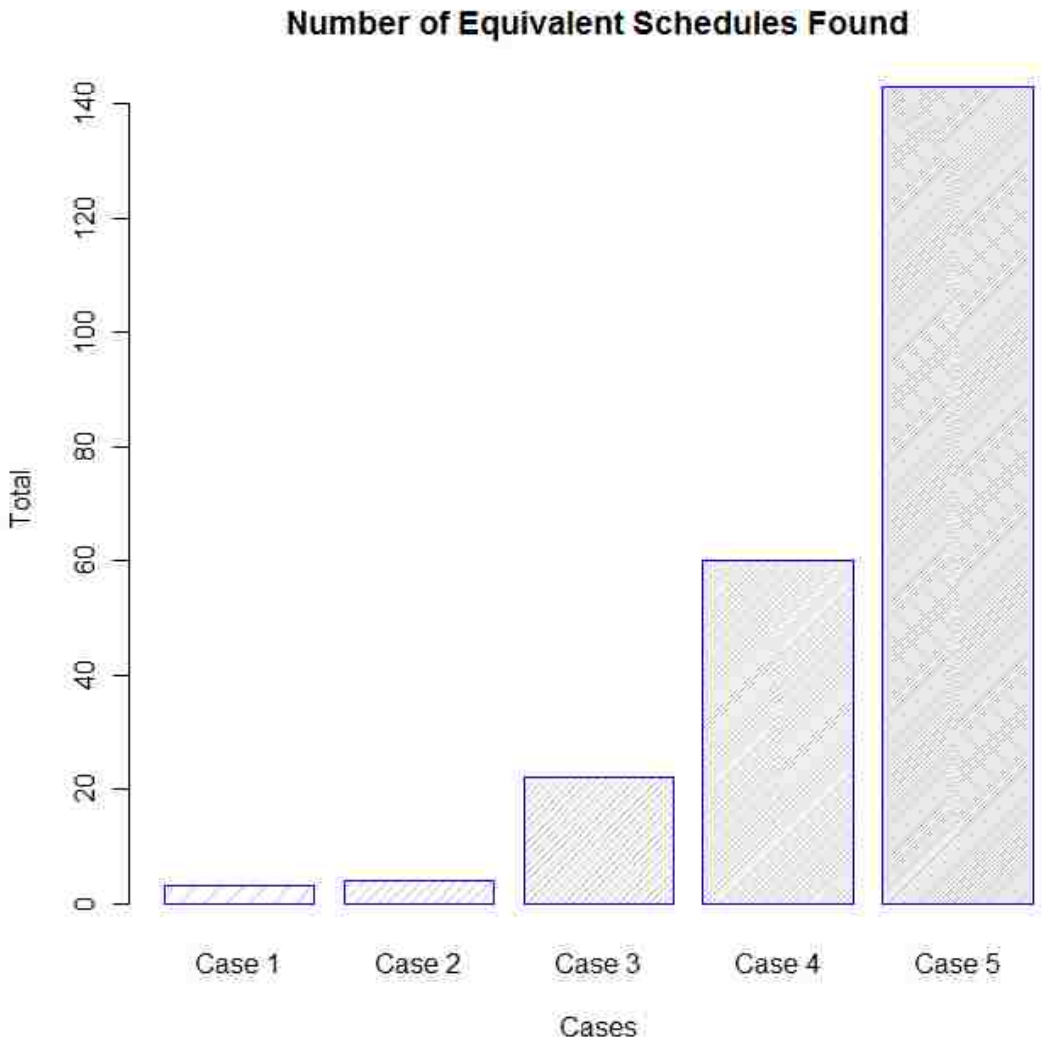


Figure 49: Graph showing the number of equivalent schedules found for each of the first 5 cases

my equivalent class algorithm I came up with a total of 2 schedules, this means the algorithm achieved an improvement of 66.666% over the legal schedules and a total improvement of 99.5238% over all possible schedules.

From these 6 cases we can begin to reach an understanding of the growth rate of the total number of legal schedules. When we compare the totals of each case in Figure 51

we can see that the total schedules grows with respect to the complexity of the threads being considered. Consider cases 4 and 5 above, even though the total number of blocks

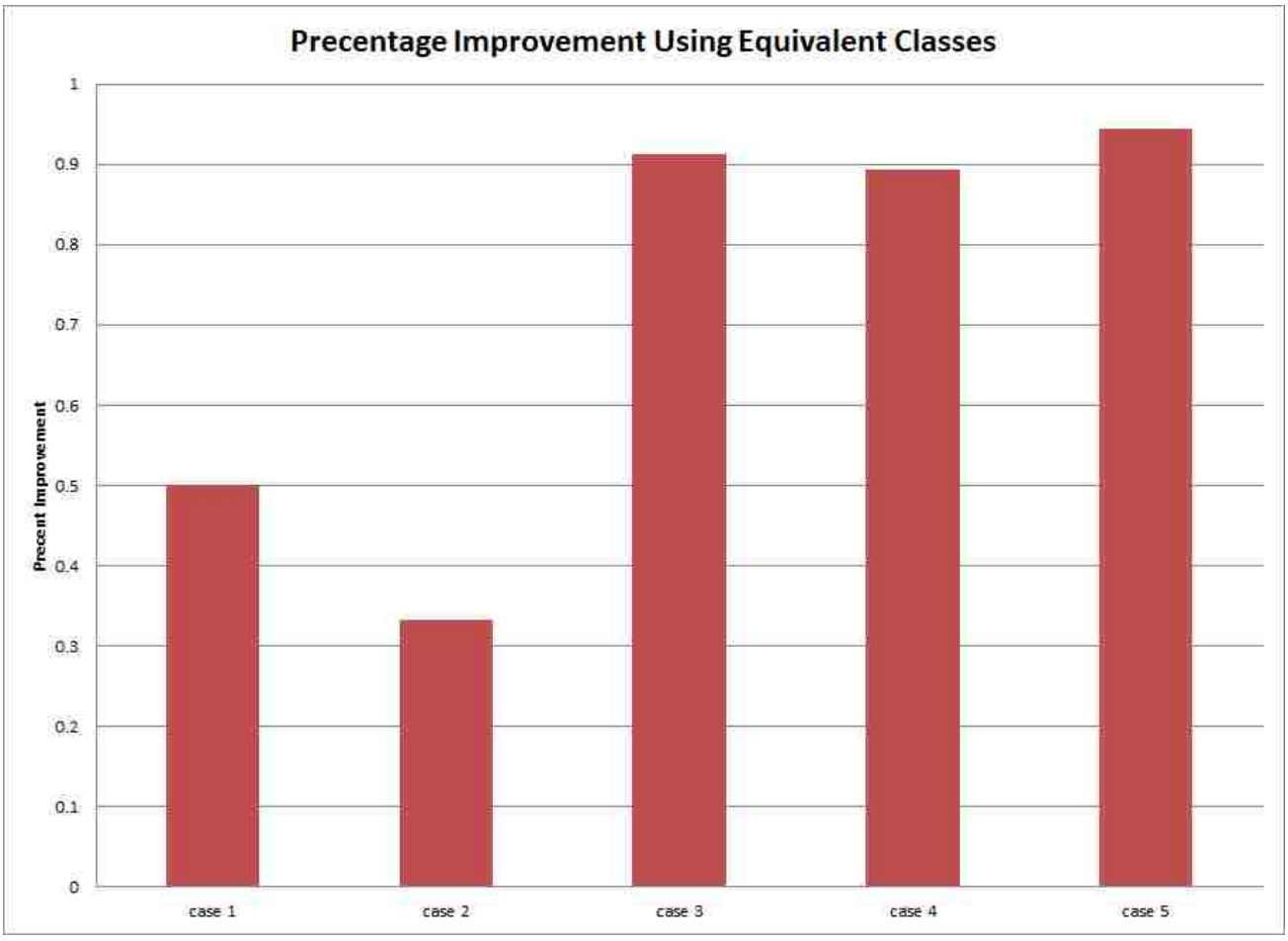


Figure 50: Graph showing the percent of improvements for each of the first 5 cases

is kept the same the total number of schedules in case 5 is around 4.5 times greater than in case 4 even though the only thing to change is that case 5 has 1 additional thread and the number of blocks per thread has been even out.

To help gain a better understanding on how the total number of legal schedules grows I conducted a series of experiments where I ran complex cases through my Perl script so that the total number of legal schedules could be found. As in case 1 through 5 above I did not take the changing state into account, I included the results of case 6 but only the

part that did not take into account the time aspect of that problem. For these tests I wanted to compare the number of blocks, the number of threads and how many legal schedules where derived using that data. The 17 cases I used are displayed in Table 1

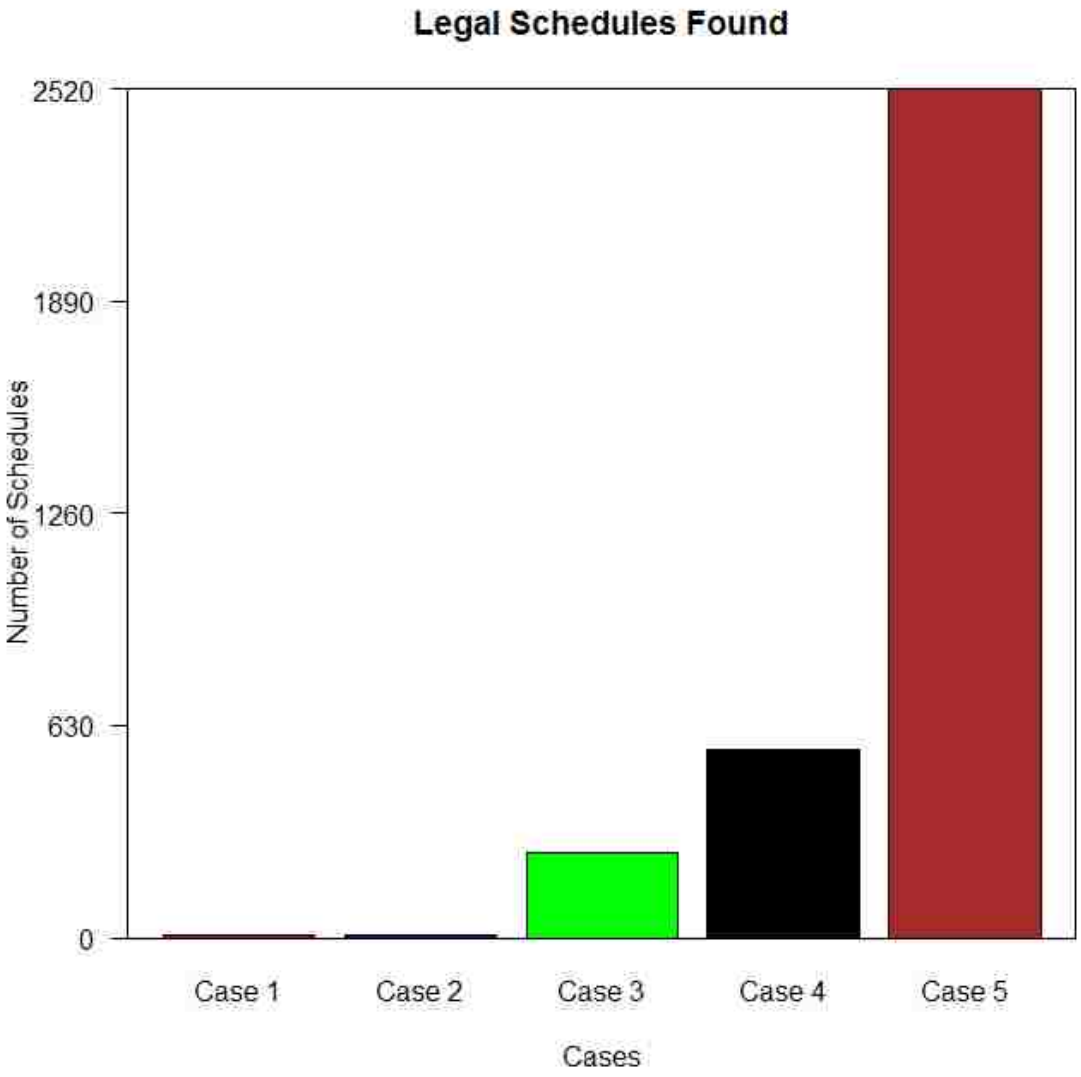


Figure 51: Graph showing the number of equivalent schedules found for each of the first 5 cases

below. As a note I included the original data from cases 1 through 6 in Table 1. The ordering of the table was based on the number of schedules from smallest to largest so the test numbers do not match the case numbers above.

As can be sees in Table 1, the total number of schedules varies greatly depending upon the combination of threads and blocks per thread. Consider test 14 and 15, in each test we have three threads and a total of 16 blocks, but the total number of schedules differs by 960,960. For test 14 there are three threads, but one thread has 8 blocks of code, the second thread has 5 and the third has 3. This combination yields a total of 720,720 schedules. On the other hand test 15 had three threads, the first 2 threads each had 6 blocks of code and the third thread had only 4 blocks. This combination yielded a total of 1,681,680.

The large difference in these two test cases highlights the fact that solving this type of problem is made more difficult by the fact that the number of schedules is that the total number of legal schedules is determined by a combination of the number of threads, the number of total blocks and how many blocks are in each thread. The combination of these three makes it very difficult to predict how many schedules any one program will have and makes it extremely difficult to determine a general rate of growth for parallel programs.

Conclusion

This paper has attempted to demonstrate that it is possible to design a general algorithm that could not only effectively find equivalent class schedules for a parallel program but would also decrease the total number of schedules to be tested so that a complete test of a parallel program would be possible. In addition it showed that the concept of equivalent classes for schedules of parallel programs is a valid method of dealing with the complexity of the scheduling problem.

Consider the 17 tests as present in Table 1. Tests 1 and 2 are easy to do a complete test on. Table 1 illustrats that complex programs are difficult to completely test. This problem is made even more difficult due to the fact that the schedules in Table 1 have already

Test	Number Threads	Number of Blocks	Number of Schedules
1	2	4	6
2	2	4	6
3	2	10	252
4	3	8	420
5	3	8	560
6	2	12	924
7	3	9	1260
8	3	9	1680
9	4	8	2520
10	4	9	7560
11	5	10	113400
12	4	12	277200
13	5	11	415800
14	3	16	720720
15	3	16	1681680
16	6	12	7484400
17	3	18	17153135

Table 1: Results of all 17 tests

been parsed into a simplified structure where only blocks of non-critical code and blocks of critical sections remain. Given that each block may represent a few lines of code or a hundred, the number of simplified schedules listed in Table 1 is only a partial listing of all the schedules for a given program.

To help solve this problem this paper presents a unique way to eliminate some of the schedules so that the overall problem is decreased in complexity. To accomplish this goal

the concept of equivalent schedules was present. This concept means that any two schedules are equivalent if the blocks of non-conflicting code between the conflicting sections are simply permutations of one and another. Secondly, the location of the conflicting schedules must be the same.

Once these conditions are met this paper showed that the total amount of improvement ranged from a little as 40% for simple conditions up to 95% for more complex setups. In addition, the experiments that were presented in this paper showed that the complexity of the problem is related to not only the total number of threads but also to the number of blocks of code, both non-critical and critical, and the number of blocks per thread.

The realization that the complexity of the scheduling problem related to those three elements helps to demonstrate the need for best practice programming techniques. As the examples in Table 1 demonstrate changing the number of blocks and the number of threads only slightly changes the overall complexity of the program. It is the combination of how the blocks of code a distributed across all threads and the number of conflicting sections that greatly changes the total number of legal schedule.

In addition to showing that the concept of equivalent classes is important for solving the overall problem, this paper also demonstrates the fact that taking into account the time based state of the program also provides a high level of improvement over checking all legal schedules. This improvement can also reduce greatly the number of schedules that must be tested by eliminate any that will never and can never be formed.

By combining all of these concepts this paper demonstrates that a general algorithm can be designed that will not only eliminate equivalent schedules but also limit the examination to schedules that are possible. These approaches demonstrate that the

potential #P-hard problem of finding all schedules of a parallel program can be reduced to a small subset of schedules that can then be easily tested for completeness. This concept relies on the fact that modern operating systems are capable of interleaving blocks of code effectively.

In this paper I presented my work on examining the difficultly of preforming a complete test on parallel programs. To solve this problem I presented the concept of using equivalent class schedules to decrease the number of total schedules to test. I then incorporated that concept into a basic algorithm for finding and all equivalent schedules of a parallel program. Additionally, I showed that by taking into account the time based component of a parallel program even more saving can be achieved with regards to the number of schedules to test and overall complexity of the problem.

Future Work

While this work was able to answer the largest of the initial questions, whether an algorithm could be designed to find equivalent class schedules, it still remains to be seen whether such an algorithm could be implemented for practical application. In addition, a related problem of determining the overall rate of improvement still remains. Due to the difficulty of determining all legal schedules and the fact that there is, as yet, no equation that describes how many equivalent schedules may exists in any one parallel program it has not yet been achieved.

To solve these remaining problems some additional work is needed. First, a large amount of data needs to be gathered so that a potential growth rate in the overall scheduling problem can be calculated. This growth rate can be found by utilizing some synthetic data to represent the complexity of the scheduling problem as was the case with

the experiments presented in this paper. This growth rate is needed so that an overall complexity of the presented algorithm can be calculated. Once the complexity is determined then it may be possible to predict the total number of equivalent classes that can be found for any given parallel program.

Works Cited

1. *Compile-time support for efficient data race detection in shared-memory parallel programs.* **Mellor-Crummey, John.** New York, NY, USA : ACM, 1993. In Proceedings of the 1993 ACM/ONR workshop on Parallel and distributed debugging (PADD '93). pp. 129-139. http://doi.acm.org/10.1145/174266.171370.

2. *On-the-fly detection of data races for programs with nested fork-join parallelism.* **Mellor-Crummey, John.** New York, NY, USA : ACM, 1991. In Proceedings of the 1991 ACM/IEEE conference on Supercomputing (Supercomputing '91). pp. 24-33.

3. *On-the-fly detection of access anomalies.* **Schonberg, D.** New York, NY, USA : ACM, 1989. In Proceedings of the ACM SIGPLAN 1989 Conference on Programming language design and implementation (PLDI '89). pp. 285-297.

4. *Automatic detection of nondeterminacy in parallel programs.* **Emrath, Perry A. and Padua, David A.** [ed.] Richard L. Wexelbalt. New York, NY, USA : ACM, 1988. In Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging (PADD '88). pp. 89-99.

5. *Fast and accurate static data-race detection for concurrent programs.* **Kahlon, Vineet, et al., et al.** [ed.] Werner Damm and Holger Hermanns. Berlin, Germany : Springer-Verlag, 2007. In Proceedings of the 19th international conference on Computer aided verification (CAV'07). pp. 226-239.

6. *RacerX: effective, static detection of race conditions and deadlocks.* **Engler, Dawson and Ashcraft, Ken.** New York, NY, USA : ACM, 2003. In Proceedings of the nineteenth ACM symposium on Operating systems principles (SOSP '03). pp. 237-252.

7. *A static analyzer for finding dynamic programming errors.* **Bush, William R., Pincus, Jonathan D. and Sielaf, David J.** 7, New York, NY, USA : John Wiley & Sons, Inc, June 2000, Software—Practice & Experience, Vol. 30, pp. 775-802.

8. *Detecting data races in Cilk programs that use locks.* **Cheng, Guang-Ien, et al., et al.** New York, NY, USA : ACM, 1998. In Proceedings of the tenth annual ACM symposium on Parallel algorithms and architectures (SPAA '98). pp. 298-309.

9. *Efficient on-the-fly data race detection in multithreaded C++ programs.* **Pozniansky, Eli and Schuster, Assaf.** New york, NY, USA : ACM, 2003. In Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming (PPoPP '03). pp. 179-190. http://doi.acm.org/10.1145/781498.781529.

10. *A type and effect system for deadlock avoidance in low-level languages.* **Gerakios, Prodromos, Papaspyrou, Nikolaos and Sagonas, Kostis.** New York, NY, USA : ACM, 2011. In Proceedings of the 7th ACM SIGPLAN workshop on Types in language design and implementation (TLDI '11). pp. 15-28.

11. *System Deadlocks.* **Coffman, E. G., Elphick, M. and Shoshani, A.** 2, New York, NY, USA : ACM, June 1971, ACM Computing Surveys, Vol. 3, pp. 67-78.

12. *The classification of deadlock prevention and avoidance is erroneous.* **Levine, Gertrude Neuman.** 2, New York, NY, USA : ACM, April 2005, ACM SIGOPS Operating Systems Review, Vol. 39, pp. 47-50.

13. *Race directed random testing of concurrent programs.* **Sen, Koushik.** New York, NY, USA : ACM, 2008. In Proceedings of the 2008 ACM SIGPLAN conference on Programming language design and implementation (PLDI '08). pp. 11-21.

14. *Randomized active atomicity violation detection in concurrent programs.* **Park, Chang-Seo and Sen, Koushik.** New york, NY, USA : ACM, 2008. In Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering (SIGSOFT '08/FSE-16). pp. 135-145.

15. *Effective random testing of concurrent programs.* **Sen, Koushik.** New York, NY, USA : ACM, 2007. In Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering (ASE '07). pp. 323-332. http://doi.acm.org/10.1145/1321631.1321679.

16. *Faster random generation of linear extensions.* **Bubley, Russ and Dyer, Martin.** Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 1998. In Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms (SODA '98). pp. 350-354.

17. *Markov chains for linear extensions, the two-dimensional case.* **Felsner, Stefan and Wernisch, Lorenz.** Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 1997. In Proceedings of the eighth annual ACM-SIAM symposium on Discrete algorithms (SODA '97). pp. 239-247.

18. *Counting linear extensions is #P-complete.* **Brightwell, Graham and Winkler, Peter.** New York, NY, USA : ACM, 1991. In Proceedings of the twenty-third annual ACM symposium on Theory of computing (STOC '91). pp. 175-181.

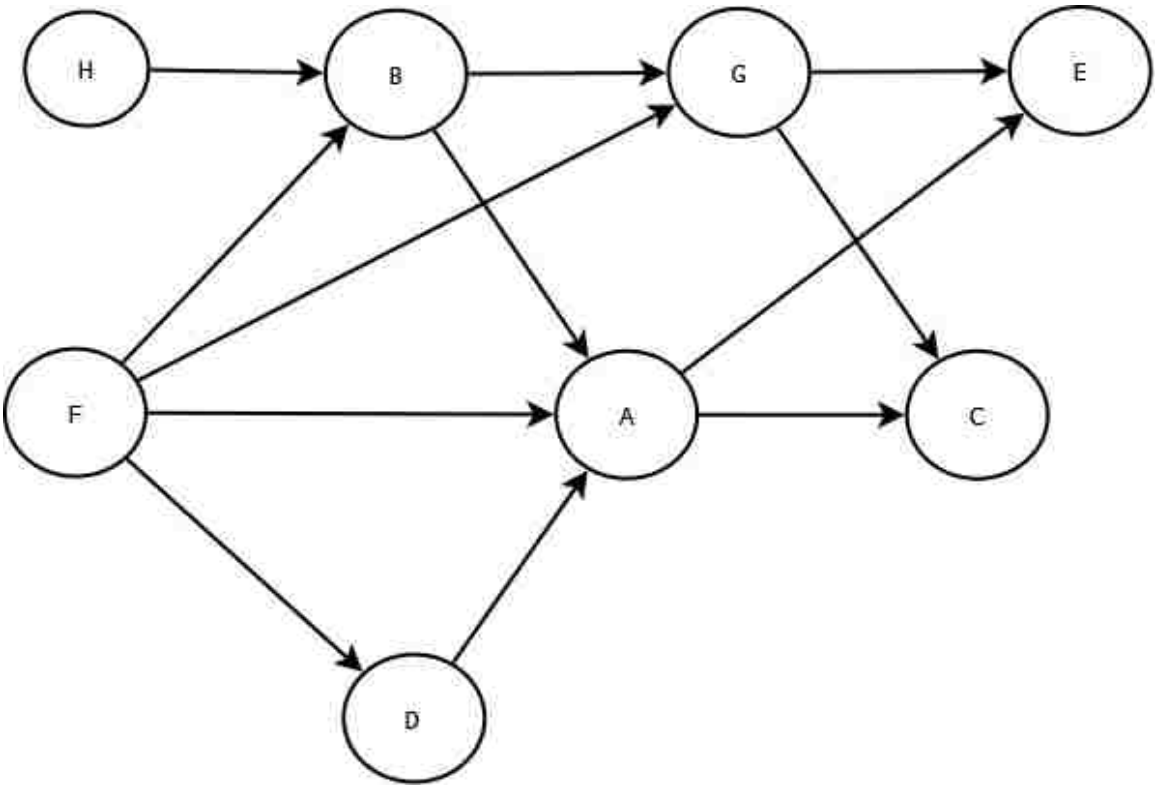
19. *I/O-efficient topological sorting of planar DAGs.* **Arge, Lars, Toma, Laura and Zeh, Norbert.** New York, NY, USA : ACM, 20039. In Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures (SPAA '03. pp. 85-93.

20. *The input/output complexity of sorting and related problems.* **Aggarwal, Alok and Jeffrey, S. Vitter.** 9, New York, NY, USA : ACM, September 1988, Communications of the ACM, Vol. 31, pp. 1116-1127.

21. **Kahn, A. B.** Topological Sorting of Large. *Communications of the ACM.* November 1962, Vol. 5, 11, pp. 558-562.

Appendix A

In A. B. Kahn's paper "Topological Sorting of Large Networks". They utilize a set of tables as the basic building block of their algorithm. For the network below the following tables will be generated. As a not this is the same example that A. B. Kahn used is his paper, only columns A,B,C and D are used in the tables in the algorithm. The other columns are used to help a person understand the table.



That network would create the table located on the next page.

Activity List

Item	Event Labels		Predecessor Flag (a)	Successor Event Location (b)
	Predecessor	Successor		
1	A	C	0	3
2	A	E	1	5
3	B	A	0	1
4	B	G	1	7
5	D	A	1	1
6	F	B	0	2
7	F	A	0	1
8	F	D	0	4
9	F	G	1	7
10	G	C	0	3
11	G	E	1	5
12	H	B	1	2

Event List

Item	Label	Count (c)	ActivityLocation (d)
1	A	3	1
2	B	2	3
3	C	2	Z
4	D	1	5
5	E	2	Z
6	F	0	6*
7	G	2	10
8	H	0	12*

Appendix B

The following code is the pseudo code for the parsing algorithm, see after the code for definition of THREAD object:

```
Input:
        S: Source files of program to be run
Code:
        ThreadArray threads[] = new ThreadArray();
        threads.addGlobalThread();
        int index = 0;
        string state = “start”;
        while(index < S.length)
        {
                if(S[index] == start of new thread)
                {
                        thread T = new thread();
                        T.happensAfter = state;
                        while(S[index] != end of thread)
                        {
                                if(S[index] == critical section)
                                {
                                        T.addCS(S[index], state);
                                }
                                else
                                {
                                        T.addNonCS(S[index], state);
                                }
                                index++;
                        }
                        threads.addThread(T);
                }
                else if(S[index] == part of Global section)
                {
                        if(S[index] == critical section)
                        {
                                threads.GlobalThreadaddCS(S[index], state);
                        }
                        else
                        {
                                threads.GlobalThreadaddNonCS (S[index], state);
                        }
                        if(S[index] == synchronization point)
                        {
                                state = threads.GlobalThreadLastBlock();
                                threads.GlobalThreadUpdateState();
```

```
                    }
                    index++;
                }
            }
```

thread Object

thread[index]

returns item at that location

thread.length

returns the length of the thread

thread.happensAfter

state variable controls when this thread may be executed

thread.addCS

add new critical section to the thread array

thread.addNonCS

add new non-critical section to the thread array

thread.addGlobalThread

add the global section to the thread array

thread.GlobalThreadaddCS

add new critical section to the global section of the thread

thread.GlobalThreadaddNonCS

add new non-critical section to the global section of the thread

thread.GlobalThreadLastBlock

returns state of last block added to global section

threads.GlobalThreadUpdateState

updates global section state

Appendix C

This is the main function for the schedule finding algorithm. See after the code for definition of SEQ object:

```
Input:
        T; set of all threads
        numT; number of threads
        fS; set of all final schedules
Code:

        for(i = 0 to i=(numT-1))
        {
                string state = "start";
                if(T[i][0].UpdateState())
                {
                        state = T[i][0].NewState();
                }

                if(T[i][0].happensAfter() == state)
                {
                        seq = new seq()

                        index[] = new array(numT); set to all Zero
                        updateIndex(index, i, T[i].length)
                        if(T[i][0] == critical section && !seq.conflicts(T[i][0]))
                        {
                                seq.add(T[i][0]);
                        }
                        else
                        {
                                seq.addBlock(T[i][0])
                        }

                        for(j = 0 to j =(numT-1) )
                        {

                                buildTree(T, numT, fS, seq, j, index, state);

                        }
                }
        }
```

seq Object

seq.add(critical section)

If the last element is a block of non-conflicting sections then close that block and add the passed in critical section to the list

seq.addBlock(non-critical section)

If the last element is a block of non-conflicting sections then add the passed in non-conflicting section to it.

Else if the last element is a conflicting critical sections then add a new block of non-conflicting section to the list and add the passed in non-critical section to that.

seqA.CSOrder == seqB.CSOrder

Returns true if the order of the conflicting critical sections in seqA is the same as they are in seqB

Returns false if not

seqA.Blocks == seqB.Blocks

Returns true if the blocks of non-conflicting sections in seqA are comprised of the same blocks of code as the non- conflicting sections in seqB and they are permutations of each other

Returns false if not

seq.conflicts(critical section)

Returns true if the passed in critical section conflicts with any other critical section already in the sequence

Returns false if not

Appendix D

The following code is the pseudo code for the build tree part of the algorithm. See after the code for definition of the prune function, updateIndex function and moreToDO function:

```
Input:
        T; set of all threads
        numT; number of threads
        fS; set of all final schedules
        seq; current partial Schedule
        tN; thread index to add node from
        index; array of current index for each thread
        state; string that holds current state for schedule
Code:

        threadIndex = index[tN]
        exit = false
        if(T[tN][threadIndex].happensAfter() == state)
        {
                if(T[tN][ threadIndex].UpdateState())
                {
                        state = T[tN][ threadIndex].NewState();
                }
                if(T[tN][threadIndex] == critical section &&
!seq.conflicts(T[tN][threadIndex]))
                {
                        seq.add(T[tN][threadIndex])
                        exit = prune(seq, fS)
                }
                else
                {
                        seq.addBlock(T[tN][threadIndex])
                }

                if(!exit)
                {
                        updateIndex (index, tN, T[tN].length)
                        if(moreToDo(index))
                        {
                                for(j = 0 to j =(numT-1))
                                {
                                        if(index[j] != -1)
                                        {
                                                buildTree(T, numT, fS, seq, j, index,
                                        state);
```

```
                                        }
                                }
                        }
                        else
                        {
                                if(!prune(seq, fS))
                                {
                                        fS.add(seq);
                                }
                        }

                }
        }
```

prune

Input:

```
        seq; current partial Schedule
        fS; set of all final schedules
```

Code:

```
        for(i = 0 to i = fS.length-1)
        {
                if(seq.CSOrder == fS[i].CSOrder && seq.Blocks == fS[i].Blocks)
                {
                        return true;
                }
        }
```

updateIndex

Input:

```
        index; array of current index for each thread
        currentThread; current thread working with
        length; length of current thread
```

Code:

```
        if(index[currentThread] +1 < length)
        {
                index[currentThread] += 1
        }
        else
        {
                index[currentThread] = -1
        }
```

moreToDo

Input:

index; array of current index for each thread

Code:

return index.Sum(); sum up all entries in the array index

Appendix E

Complete list of all permutations of first equivalent class for initial walkthrough

BC1,CS1a,BC2,BC4,CS4c,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS5d,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS5d,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC4,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC4,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC5,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS4c,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS5d,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS5d,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS4c,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC4,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC4,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC5,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS4c,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS5d,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS5d,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS4c,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d

BC1,BC4,CS4c,CS1a,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS4c,CS5d,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC2,BC4,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC2,BC4,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC2,BC5,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC5,BC4,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d

BC1,CS5d,BC5,BC4,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS5d,BC5,BC4,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS4c,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS5d,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS5d,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS4c,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS1a,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS4c,CS5d,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS1a,BC2,CS5d,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,BC1,CS5d,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,CS5d,BC1,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS4c,CS5d,BC5,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d

BC4,CS5d,BC1,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,CS4c,BC1,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,CS4c,BC5,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC4,CS5d,BC5,CS4c,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC4,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC4,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC5,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d

CS5d,BC4,BC1,CS1a,BC2,CS4c,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC2,BC5,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC5,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC5,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,CS4c,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC2,BC5,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC5,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,CS4c,BC1,BC5,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,CS4c,BC5,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC4,BC5,CS4c,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,CS1a,BC2,BC4,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,CS1a,BC4,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,CS1a,BC4,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS1a,BC2,CS4c,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS1a,CS4c,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS4c,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
CS5d,BC5,BC4,CS4c,BC1,CS1a,BC2,CS1b,CS2a,BC3,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS4c,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS5d,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS5d,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC4,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC4,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,CS5d,BC5,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d

BC1,CS1a,BC4,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC4,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC4,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC4,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC2,BC5,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC4,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,CS5d,BC5,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS1a,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS4c,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,BC4,CS5d,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC2,BC4,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC2,BC4,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d

BC1,CS5d,CS1a,BC2,BC5,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC4,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,CS1a,BC5,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC4,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS5d,BC5,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS1a,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS4c,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d

BC4,BC1,CS5d,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,BC1,CS5d,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,BC1,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,CS5d,BC1,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS4c,CS5d,BC5,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC1,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,CS4c,BC1,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,CS4c,BC5,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC5,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC4,CS5d,BC5,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC4,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC4,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC2,BC5,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d

CS5d,BC1,CS1a,BC4,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC4,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,CS1a,BC5,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC4,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC1,BC5,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC1,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC2,BC5,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC5,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,CS4c,BC1,BC5,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,CS4c,BC5,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC5,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC4,BC5,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d

CS5d,BC5,BC1,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC1,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC1,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC1,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC4,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
CS5d,BC5,BC4,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS2a,CS3b,CS1d
BC1,CS1a,BC2,BC4,CS4c,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC2,BC4,CS5d,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC2,BC4,CS5d,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC2,CS5d,BC4,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC2,CS5d,BC4,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC2,CS5d,BC5,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,BC4,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC2,BC4,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC2,BC4,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC2,BC5,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC4,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC5,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC5,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS1a,CS5d,BC5,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d

BC1,BC4,CS1a,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS1a,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS4c,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,BC4,CS5d,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC2,BC4,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC2,BC4,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC2,BC5,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC4,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC5,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC5,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,CS1a,BC5,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d

BC1,CS5d,BC4,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC4,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC1,CS5d,BC5,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,BC2,CS4c,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,BC2,CS5d,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,BC2,CS5d,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS4c,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS4c,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS1a,CS5d,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS4c,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,BC1,CS5d,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS1a,BC2,CS5d,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS1a,CS5d,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS5d,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,BC1,CS5d,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,CS5d,BC1,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d

BC4,CS4c,CS5d,BC1,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS4c,CS5d,BC5,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC1,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,CS4c,BC1,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,CS4c,BC1,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,CS4c,BC5,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC5,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC5,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC5,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
BC4,CS5d,BC5,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC2,BC4,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC2,BC4,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC2,BC5,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC4,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC5,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC5,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,CS1a,BC5,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC4,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d

CS5d,BC1,BC5,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC5,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC5,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC5,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC5,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC1,BC5,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,BC2,CS4c,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,BC2,BC5,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,CS4c,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,BC5,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS1a,BC5,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS4c,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,CS4c,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,BC5,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,BC5,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC1,BC5,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC2,BC5,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,CS4c,BC1,CS1a,BC5,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,CS4c,BC1,BC5,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,CS4c,BC5,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC5,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC5,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC5,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC4,BC5,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,CS1a,BC2,BC4,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,CS1a,BC4,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,CS1a,BC4,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,BC4,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,BC4,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC1,BC4,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC4,BC1,CS1a,BC2,CS4c,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC4,BC1,CS1a,CS4c,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC4,BC1,CS4c,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d
CS5d,BC5,BC4,CS4c,BC1,CS1a,BC2,CS1b,BC3,CS3b,CS2a,CS1d

Appendix F

All legal Schedules for Case 3:

BC1,CS2a,BC2,CS3a,BC3,BC4,CS1b,CS2b,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,BC3,CS1b,CS2b,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC2,CS3a,BC4,CS1b,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC2,BC4,CS3a,BC3,CS1b,CS2b,CS3b,BC5
BC1,CS2a,BC2,BC4,CS3a,CS1b,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC2,BC4,CS3a,CS1b,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC2,BC4,CS3a,CS1b,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC2,BC4,CS3a,CS1b,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC2,BC4,CS1b,CS3a,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC2,BC4,CS1b,CS3a,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC2,BC4,CS1b,CS3a,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS3a,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3a,CS3b,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3a,CS3b,BC5,BC3
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3b,CS3a,BC5,BC3
BC1,CS2a,BC2,BC4,CS1b,CS2b,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,BC2,CS3a,BC3,CS1b,CS2b,CS3b,BC5
BC1,CS2a,BC4,BC2,CS3a,CS1b,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC4,BC2,CS3a,CS1b,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC4,BC2,CS3a,CS1b,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC4,BC2,CS3a,CS1b,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC4,BC2,CS1b,CS3a,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC4,BC2,CS1b,CS3a,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC4,BC2,CS1b,CS3a,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC4,BC2,CS1b,CS3a,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3a,CS3b,BC3,BC5
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3a,CS3b,BC5,BC3
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3b,CS3a,BC5,BC3
BC1,CS2a,BC4,BC2,CS1b,CS2b,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC1,CS2a,BC4,CS1b,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC1,CS2a,BC4,CS1b,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC1,CS2a,BC4,CS1b,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3a,CS3b,BC3,BC5

BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC1,CS2a,BC4,CS1b,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC1,CS2a,BC4,CS1b,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC1,CS2a,BC4,CS1b,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC1,BC4,CS2a,BC2,CS3a,BC3,CS1b,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,BC3,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,BC3,CS3b,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC3,BC5
BC1,BC4,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS3a,BC3,CS2b,CS3b,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,BC3,CS3b,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC3,BC5
BC1,BC4,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3a,BC3,CS3b,BC5
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3a,CS3b,BC3,BC5
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3a,CS3b,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3b,CS3a,BC3,BC5
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3b,CS3a,BC5,BC3
BC1,BC4,CS2a,BC2,CS1b,CS2b,CS3b,BC5,CS3a,BC3
BC1,BC4,CS2a,CS1b,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC1,BC4,CS2a,CS1b,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC1,BC4,CS2a,CS1b,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC1,BC4,CS2a,CS1b,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3a,CS3b,BC3,BC5
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC1,BC4,CS2a,CS1b,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC1,BC4,CS2a,CS1b,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC1,BC4,CS2a,CS1b,CS2b,CS3b,BC2,CS3a,BC3,BC5

BC1,BC4,CS2a,CS1b,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC1,BC4,CS2a,CS1b,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC1,BC4,CS2a,CS1b,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC1,BC4,CS1b,CS2a,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC1,BC4,CS1b,CS2a,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC1,BC4,CS1b,CS2a,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC1,BC4,CS1b,CS2a,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3a,CS3b,BC3,BC5
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2a,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2a,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2a,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2a,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2a,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2a,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3a,BC3,CS3b,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3a,CS3b,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3a,CS3b,BC5,BC3
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3b,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3b,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2b,CS2a,BC2,CS3b,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC2,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC2,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS2a,CS3b,BC5,BC2,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC2,CS3a,BC3,BC5
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC2,CS3a,BC5,BC3
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC2,BC5,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS3b,CS2a,BC5,BC2,CS3a,BC3
BC1,BC4,CS1b,CS2b,CS3b,BC5,CS2a,BC2,CS3a,BC3
BC4,BC1,CS2a,BC2,CS3a,BC3,CS1b,CS2b,CS3b,BC5
BC4,BC1,CS2a,BC2,CS3a,CS1b,BC3,CS2b,CS3b,BC5
BC4,BC1,CS2a,BC2,CS3a,CS1b,CS2b,BC3,CS3b,BC5
BC4,BC1,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC3,BC5
BC4,BC1,CS2a,BC2,CS3a,CS1b,CS2b,CS3b,BC5,BC3
BC4,BC1,CS2a,BC2,CS1b,CS3a,BC3,CS2b,CS3b,BC5
BC4,BC1,CS2a,BC2,CS1b,CS3a,CS2b,BC3,CS3b,BC5
BC4,BC1,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC3,BC5

BC4,BC1,CS2a,BC2,CS1b,CS3a,CS2b,CS3b,BC5,BC3
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3a,BC3,CS3b,BC5
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3a,CS3b,BC3,BC5
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3a,CS3b,BC5,BC3
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3b,CS3a,BC3,BC5
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3b,CS3a,BC5,BC3
BC4,BC1,CS2a,BC2,CS1b,CS2b,CS3b,BC5,CS3a,BC3
BC4,BC1,CS2a,CS1b,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC4,BC1,CS2a,CS1b,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC4,BC1,CS2a,CS1b,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC4,BC1,CS2a,CS1b,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3a,CS3b,BC3,BC5
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC4,BC1,CS2a,CS1b,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC4,BC1,CS2a,CS1b,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC4,BC1,CS2a,CS1b,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC4,BC1,CS2a,CS1b,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC4,BC1,CS2a,CS1b,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC4,BC1,CS2a,CS1b,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC4,BC1,CS1b,CS2a,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC4,BC1,CS1b,CS2a,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC4,BC1,CS1b,CS2a,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC4,BC1,CS1b,CS2a,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3a,CS3b,BC3,BC5
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2a,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2a,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC4,BC1,CS1b,CS2a,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2a,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2a,CS2b,CS3b,BC2,BC5,CS3a,BC3

BC4,BC1,CS1b,CS2a,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3a,BC3,CS3b,BC5
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3a,CS3b,BC3,BC5
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3a,CS3b,BC5,BC3
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3b,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3b,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2b,CS2a,BC2,CS3b,BC5,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS2a,CS3b,BC2,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2b,CS2a,CS3b,BC2,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2b,CS2a,CS3b,BC2,BC5,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS2a,CS3b,BC5,BC2,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS3b,CS2a,BC2,CS3a,BC3,BC5
BC4,BC1,CS1b,CS2b,CS3b,CS2a,BC2,CS3a,BC5,BC3
BC4,BC1,CS1b,CS2b,CS3b,CS2a,BC2,BC5,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS3b,CS2a,BC5,BC2,CS3a,BC3
BC4,BC1,CS1b,CS2b,CS3b,BC5,CS2a,BC2,CS3a,BC3
BC4,CS1b,BC1,CS2a,BC2,CS3a,BC3,CS2b,CS3b,BC5
BC4,CS1b,BC1,CS2a,BC2,CS3a,CS2b,BC3,CS3b,BC5
BC4,CS1b,BC1,CS2a,BC2,CS3a,CS2b,CS3b,BC3,BC5
BC4,CS1b,BC1,CS2a,BC2,CS3a,CS2b,CS3b,BC5,BC3
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3a,BC3,CS3b,BC5
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3a,CS3b,BC3,BC5
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3a,CS3b,BC5,BC3
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3b,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3b,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2a,BC2,CS2b,CS3b,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3a,BC3,CS3b,BC5
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3a,CS3b,BC3,BC5
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3a,CS3b,BC5,BC3
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3b,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3b,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2a,CS2b,BC2,CS3b,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2a,CS2b,CS3b,BC2,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2a,CS2b,CS3b,BC2,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2a,CS2b,CS3b,BC2,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2a,CS2b,CS3b,BC5,BC2,CS3a,BC3
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3a,BC3,CS3b,BC5
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3a,CS3b,BC3,BC5
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3a,CS3b,BC5,BC3
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3b,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3b,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2b,CS2a,BC2,CS3b,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2b,CS2a,CS3b,BC2,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2b,CS2a,CS3b,BC2,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2b,CS2a,CS3b,BC2,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2b,CS2a,CS3b,BC5,BC2,CS3a,BC3

BC4,CS1b,BC1,CS2b,CS3b,CS2a,BC2,CS3a,BC3,BC5
BC4,CS1b,BC1,CS2b,CS3b,CS2a,BC2,CS3a,BC5,BC3
BC4,CS1b,BC1,CS2b,CS3b,CS2a,BC2,BC5,CS3a,BC3
BC4,CS1b,BC1,CS2b,CS3b,CS2a,BC5,BC2,CS3a,BC3
BC4,CS1b,BC1,CS2b,CS3b,BC5,CS2a,BC2,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3a,BC3,CS3b,BC5
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3a,CS3b,BC3,BC5
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3a,CS3b,BC5,BC3
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3b,CS3a,BC3,BC5
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3b,CS3a,BC5,BC3
BC4,CS1b,CS2b,BC1,CS2a,BC2,CS3b,BC5,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS2a,CS3b,BC2,CS3a,BC3,BC5
BC4,CS1b,CS2b,BC1,CS2a,CS3b,BC2,CS3a,BC5,BC3
BC4,CS1b,CS2b,BC1,CS2a,CS3b,BC2,BC5,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS2a,CS3b,BC5,BC2,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS3b,CS2a,BC2,CS3a,BC3,BC5
BC4,CS1b,CS2b,BC1,CS3b,CS2a,BC2,CS3a,BC5,BC3
BC4,CS1b,CS2b,BC1,CS3b,CS2a,BC2,BC5,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS3b,CS2a,BC5,BC2,CS3a,BC3
BC4,CS1b,CS2b,BC1,CS3b,BC5,CS2a,BC2,CS3a,BC3
BC4,CS1b,CS2b,CS3b,BC1,CS2a,BC2,CS3a,BC3,BC5
BC4,CS1b,CS2b,CS3b,BC1,CS2a,BC2,CS3a,BC5,BC3
BC4,CS1b,CS2b,CS3b,BC1,CS2a,BC2,BC5,CS3a,BC3
BC4,CS1b,CS2b,CS3b,BC1,CS2a,BC5,BC2,CS3a,BC3
BC4,CS1b,CS2b,CS3b,BC1,BC5,CS2a,BC2,CS3a,BC3
BC4,CS1b,CS2b,CS3b,BC5,BC1,CS2a,BC2,CS3a,BC3

Appendix G

Legal schedules for Case 4:

BC1,CS2a,BC2,CS3b,BC3,CS2d,CS3d,BC5
BC1,CS2a,BC2,CS3b,CS2d,BC3,CS3d,BC5
BC1,CS2a,BC2,CS3b,CS2d,CS3d,BC3,BC5
BC1,CS2a,BC2,CS3b,CS2d,CS3d,BC5,BC3
BC1,CS2a,BC2,CS2d,CS3b,BC3,CS3d,BC5
BC1,CS2a,BC2,CS2d,CS3b,CS3d,BC3,BC5
BC1,CS2a,BC2,CS2d,CS3b,CS3d,BC5,BC3
BC1,CS2a,BC2,CS2d,CS3d,CS3b,BC3,BC5
BC1,CS2a,BC2,CS2d,CS3d,CS3b,BC5,BC3
BC1,CS2a,BC2,CS2d,CS3d,BC5,CS3b,BC3
BC1,CS2a,CS2d,BC2,CS3b,BC3,CS3d,BC5
BC1,CS2a,CS2d,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3b,CS3d,BC5,BC3
BC1,CS2a,CS2d,BC2,CS3d,CS3b,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3d,CS3b,BC5,BC3
BC1,CS2a,CS2d,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2a,CS2d,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2a,CS2d,CS3d,BC2,CS3b,BC5,BC3
BC1,CS2a,CS2d,CS3d,BC2,BC5,CS3b,BC3
BC1,CS2a,CS2d,CS3d,BC5,BC2,CS3b,BC3
BC1,BC2,CS2a,CS3b,BC3,CS2d,CS3d,BC5
BC1,BC2,CS2a,CS3b,CS2d,BC3,CS3d,BC5
BC1,BC2,CS2a,CS3b,CS2d,CS3d,BC3,BC5
BC1,BC2,CS2a,CS3b,CS2d,CS3d,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3b,BC3,CS3d,BC5
BC1,BC2,CS2a,CS2d,CS3b,CS3d,BC3,BC5
BC1,BC2,CS2a,CS2d,CS3b,CS3d,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3d,CS3b,BC3,BC5
BC1,BC2,CS2a,CS2d,CS3d,CS3b,BC5,BC3
BC1,BC2,CS2a,CS2d,CS3d,BC5,CS3b,BC3
BC1,BC2,CS3b,CS2a,BC3,CS2d,CS3d,BC5
BC1,BC2,CS3b,CS2a,CS2d,BC3,CS3d,BC5
BC1,BC2,CS3b,CS2a,CS2d,CS3d,BC3,BC5
BC1,BC2,CS3b,CS2a,CS2d,CS3d,BC5,BC3
BC1,BC2,CS3b,BC3,CS2a,CS2d,CS3d,BC5
BC1,BC2,CS3b,BC3,CS2d,CS2a,CS3d,BC5
BC1,BC2,CS3b,BC3,CS2d,CS3d,CS2a,BC5
BC1,BC2,CS3b,BC3,CS2d,CS3d,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS2a,BC3,CS3d,BC5
BC1,BC2,CS3b,CS2d,CS2a,CS3d,BC3,BC5
BC1,BC2,CS3b,CS2d,CS2a,CS3d,BC5,BC3
BC1,BC2,CS3b,CS2d,BC3,CS2a,CS3d,BC5

BC1,BC2,CS3b,CS2d,BC3,CS3d,CS2a,BC5
BC1,BC2,CS3b,CS2d,BC3,CS3d,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS3d,CS2a,BC3,BC5
BC1,BC2,CS3b,CS2d,CS3d,CS2a,BC5,BC3
BC1,BC2,CS3b,CS2d,CS3d,BC3,CS2a,BC5
BC1,BC2,CS3b,CS2d,CS3d,BC3,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS3d,BC5,CS2a,BC3
BC1,BC2,CS3b,CS2d,CS3d,BC5,BC3,CS2a
BC1,BC2,CS2d,CS2a,CS3b,BC3,CS3d,BC5
BC1,BC2,CS2d,CS2a,CS3b,CS3d,BC3,BC5
BC1,BC2,CS2d,CS2a,CS3b,CS3d,BC5,BC3
BC1,BC2,CS2d,CS2a,CS3d,CS3b,BC3,BC5
BC1,BC2,CS2d,CS2a,CS3d,CS3b,BC5,BC3
BC1,BC2,CS2d,CS2a,CS3d,BC5,CS3b,BC3
BC1,BC2,CS2d,CS3b,CS2a,BC3,CS3d,BC5
BC1,BC2,CS2d,CS3b,CS2a,CS3d,BC3,BC5
BC1,BC2,CS2d,CS3b,CS2a,CS3d,BC5,BC3
BC1,BC2,CS2d,CS3b,BC3,CS2a,CS3d,BC5
BC1,BC2,CS2d,CS3b,BC3,CS3d,CS2a,BC5
BC1,BC2,CS2d,CS3b,BC3,CS3d,BC5,CS2a
BC1,BC2,CS2d,CS3b,CS3d,CS2a,BC3,BC5
BC1,BC2,CS2d,CS3b,CS3d,CS2a,BC5,BC3
BC1,BC2,CS2d,CS3b,CS3d,BC3,CS2a,BC5
BC1,BC2,CS2d,CS3b,CS3d,BC3,BC5,CS2a
BC1,BC2,CS2d,CS3b,CS3d,BC5,CS2a,BC3
BC1,BC2,CS2d,CS3b,CS3d,BC5,BC3,CS2a
BC1,BC2,CS2d,CS3d,CS2a,CS3b,BC3,BC5
BC1,BC2,CS2d,CS3d,CS2a,CS3b,BC5,BC3
BC1,BC2,CS2d,CS3d,CS2a,BC5,CS3b,BC3
BC1,BC2,CS2d,CS3d,CS3b,CS2a,BC3,BC5
BC1,BC2,CS2d,CS3d,CS3b,CS2a,BC5,BC3
BC1,BC2,CS2d,CS3d,CS3b,BC3,CS2a,BC5
BC1,BC2,CS2d,CS3d,CS3b,BC3,BC5,CS2a
BC1,BC2,CS2d,CS3d,CS3b,BC5,CS2a,BC3
BC1,BC2,CS2d,CS3d,CS3b,BC5,BC3,CS2a
BC1,BC2,CS2d,CS3d,BC5,CS2a,CS3b,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,CS2a,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,BC3,CS2a
BC1,CS2d,CS2a,BC2,CS3b,BC3,CS3d,BC5
BC1,CS2d,CS2a,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2d,CS2a,BC2,CS3b,CS3d,BC5,BC3
BC1,CS2d,CS2a,BC2,CS3d,CS3b,BC3,BC5
BC1,CS2d,CS2a,BC2,CS3d,CS3b,BC5,BC3
BC1,CS2d,CS2a,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2d,CS2a,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2d,CS2a,CS3d,BC2,CS3b,BC5,BC3

BC1,CS2d,CS2a,CS3d,BC2,BC5,CS3b,BC3
BC1,CS2d,CS2a,CS3d,BC5,BC2,CS3b,BC3
BC1,CS2d,BC2,CS2a,CS3b,BC3,CS3d,BC5
BC1,CS2d,BC2,CS2a,CS3b,CS3d,BC3,BC5
BC1,CS2d,BC2,CS2a,CS3b,CS3d,BC5,BC3
BC1,CS2d,BC2,CS2a,CS3d,CS3b,BC3,BC5
BC1,CS2d,BC2,CS2a,CS3d,CS3b,BC5,BC3
BC1,CS2d,BC2,CS2a,CS3d,BC5,CS3b,BC3
BC1,CS2d,BC2,CS3b,CS2a,BC3,CS3d,BC5
BC1,CS2d,BC2,CS3b,CS2a,CS3d,BC3,BC5
BC1,CS2d,BC2,CS3b,CS2a,CS3d,BC5,BC3
BC1,CS2d,BC2,CS3b,BC3,CS2a,CS3d,BC5
BC1,CS2d,BC2,CS3b,BC3,CS3d,CS2a,BC5
BC1,CS2d,BC2,CS3b,BC3,CS3d,BC5,CS2a
BC1,CS2d,BC2,CS3b,CS3d,CS2a,BC3,BC5
BC1,CS2d,BC2,CS3b,CS3d,CS2a,BC5,BC3
BC1,CS2d,BC2,CS3b,CS3d,BC3,CS2a,BC5
BC1,CS2d,BC2,CS3b,CS3d,BC3,BC5,CS2a
BC1,CS2d,BC2,CS3b,CS3d,BC5,CS2a,BC3
BC1,CS2d,BC2,CS3b,CS3d,BC5,BC3,CS2a
BC1,CS2d,BC2,CS3d,CS2a,CS3b,BC3,BC5
BC1,CS2d,BC2,CS3d,CS2a,CS3b,BC5,BC3
BC1,CS2d,BC2,CS3d,CS2a,BC5,CS3b,BC3
BC1,CS2d,BC2,CS3d,CS3b,CS2a,BC3,BC5
BC1,CS2d,BC2,CS3d,CS3b,CS2a,BC5,BC3
BC1,CS2d,BC2,CS3d,CS3b,BC3,CS2a,BC5
BC1,CS2d,BC2,CS3d,CS3b,BC3,BC5,CS2a
BC1,CS2d,BC2,CS3d,CS3b,BC5,CS2a,BC3
BC1,CS2d,BC2,CS3d,CS3b,BC5,BC3,CS2a
BC1,CS2d,BC2,CS3d,BC5,CS2a,CS3b,BC3
BC1,CS2d,BC2,CS3d,BC5,CS3b,CS2a,BC3
BC1,CS2d,BC2,CS3d,BC5,CS3b,BC3,CS2a
BC1,CS2d,CS3d,CS2a,BC2,CS3b,BC3,BC5
BC1,CS2d,CS3d,CS2a,BC2,CS3b,BC5,BC3
BC1,CS2d,CS3d,CS2a,BC2,BC5,CS3b,BC3
BC1,CS2d,CS3d,CS2a,BC5,BC2,CS3b,BC3
BC1,CS2d,CS3d,BC2,CS2a,CS3b,BC3,BC5
BC1,CS2d,CS3d,BC2,CS2a,CS3b,BC5,BC3
BC1,CS2d,CS3d,BC2,CS2a,BC5,CS3b,BC3
BC1,CS2d,CS3d,BC2,CS3b,CS2a,BC3,BC5
BC1,CS2d,CS3d,BC2,CS3b,CS2a,BC5,BC3
BC1,CS2d,CS3d,BC2,CS3b,BC3,CS2a,BC5
BC1,CS2d,CS3d,BC2,CS3b,BC3,BC5,CS2a
BC1,CS2d,CS3d,BC2,CS3b,BC5,CS2a,BC3
BC1,CS2d,CS3d,BC2,CS3b,BC5,BC3,CS2a
BC1,CS2d,CS3d,BC2,BC5,CS2a,CS3b,BC3

BC1,CS2d,CS3d,BC2,BC5,CS3b,CS2a,BC3
BC1,CS2d,CS3d,BC2,BC5,CS3b,BC3,CS2a
BC1,CS2d,CS3d,BC5,CS2a,BC2,CS3b,BC3
BC1,CS2d,CS3d,BC5,BC2,CS2a,CS3b,BC3
BC1,CS2d,CS3d,BC5,BC2,CS3b,CS2a,BC3
BC1,CS2d,CS3d,BC5,BC2,CS3b,BC3,CS2a
BC2,BC1,CS2a,CS3b,BC3,CS2d,CS3d,BC5
BC2,BC1,CS2a,CS3b,CS2d,BC3,CS3d,BC5
BC2,BC1,CS2a,CS3b,CS2d,CS3d,BC3,BC5
BC2,BC1,CS2a,CS3b,CS2d,CS3d,BC5,BC3
BC2,BC1,CS2a,CS2d,CS3b,BC3,CS3d,BC5
BC2,BC1,CS2a,CS2d,CS3b,CS3d,BC3,BC5
BC2,BC1,CS2a,CS2d,CS3b,CS3d,BC5,BC3
BC2,BC1,CS2a,CS2d,CS3d,CS3b,BC3,BC5
BC2,BC1,CS2a,CS2d,CS3d,CS3b,BC5,BC3
BC2,BC1,CS2a,CS2d,CS3d,BC5,CS3b,BC3
BC2,BC1,CS3b,CS2a,BC3,CS2d,CS3d,BC5
BC2,BC1,CS3b,CS2a,CS2d,BC3,CS3d,BC5
BC2,BC1,CS3b,CS2a,CS2d,CS3d,BC3,BC5
BC2,BC1,CS3b,CS2a,CS2d,CS3d,BC5,BC3
BC2,BC1,CS3b,BC3,CS2a,CS2d,CS3d,BC5
BC2,BC1,CS3b,BC3,CS2d,CS2a,CS3d,BC5
BC2,BC1,CS3b,BC3,CS2d,CS3d,CS2a,BC5
BC2,BC1,CS3b,BC3,CS2d,CS3d,BC5,CS2a
BC2,BC1,CS3b,CS2d,CS2a,BC3,CS3d,BC5
BC2,BC1,CS3b,CS2d,CS2a,CS3d,BC3,BC5
BC2,BC1,CS3b,CS2d,CS2a,CS3d,BC5,BC3
BC2,BC1,CS3b,CS2d,BC3,CS2a,CS3d,BC5
BC2,BC1,CS3b,CS2d,BC3,CS3d,CS2a,BC5
BC2,BC1,CS3b,CS2d,BC3,CS3d,BC5,CS2a
BC2,BC1,CS3b,CS2d,CS3d,CS2a,BC3,BC5
BC2,BC1,CS3b,CS2d,CS3d,CS2a,BC5,BC3
BC2,BC1,CS3b,CS2d,CS3d,BC3,CS2a,BC5
BC2,BC1,CS3b,CS2d,CS3d,BC3,BC5,CS2a
BC2,BC1,CS3b,CS2d,CS3d,BC5,CS2a,BC3
BC2,BC1,CS3b,CS2d,CS3d,BC5,BC3,CS2a
BC2,BC1,CS2d,CS2a,CS3b,BC3,CS3d,BC5
BC2,BC1,CS2d,CS2a,CS3b,CS3d,BC3,BC5
BC2,BC1,CS2d,CS2a,CS3b,CS3d,BC5,BC3
BC2,BC1,CS2d,CS2a,CS3d,CS3b,BC3,BC5
BC2,BC1,CS2d,CS2a,CS3d,CS3b,BC5,BC3
BC2,BC1,CS2d,CS2a,CS3d,BC5,CS3b,BC3
BC2,BC1,CS2d,CS3b,CS2a,BC3,CS3d,BC5
BC2,BC1,CS2d,CS3b,CS2a,CS3d,BC3,BC5
BC2,BC1,CS2d,CS3b,CS2a,CS3d,BC5,BC3
BC2,BC1,CS2d,CS3b,BC3,CS2a,CS3d,BC5

BC2,BC1,CS2d,CS3b,BC3,CS3d,CS2a,BC5
BC2,BC1,CS2d,CS3b,BC3,CS3d,BC5,CS2a
BC2,BC1,CS2d,CS3b,CS3d,CS2a,BC3,BC5
BC2,BC1,CS2d,CS3b,CS3d,CS2a,BC5,BC3
BC2,BC1,CS2d,CS3b,CS3d,BC3,CS2a,BC5
BC2,BC1,CS2d,CS3b,CS3d,BC3,BC5,CS2a
BC2,BC1,CS2d,CS3b,CS3d,BC5,CS2a,BC3
BC2,BC1,CS2d,CS3b,CS3d,BC5,BC3,CS2a
BC2,BC1,CS2d,CS3d,CS2a,CS3b,BC3,BC5
BC2,BC1,CS2d,CS3d,CS2a,CS3b,BC5,BC3
BC2,BC1,CS2d,CS3d,CS2a,BC5,CS3b,BC3
BC2,BC1,CS2d,CS3d,CS3b,CS2a,BC3,BC5
BC2,BC1,CS2d,CS3d,CS3b,CS2a,BC5,BC3
BC2,BC1,CS2d,CS3d,CS3b,BC3,CS2a,BC5
BC2,BC1,CS2d,CS3d,CS3b,BC3,BC5,CS2a
BC2,BC1,CS2d,CS3d,CS3b,BC5,CS2a,BC3
BC2,BC1,CS2d,CS3d,CS3b,BC5,BC3,CS2a
BC2,BC1,CS2d,CS3d,BC5,CS2a,CS3b,BC3
BC2,BC1,CS2d,CS3d,BC5,CS3b,CS2a,BC3
BC2,BC1,CS2d,CS3d,BC5,CS3b,BC3,CS2a
BC2,CS3b,BC1,CS2a,BC3,CS2d,CS3d,BC5
BC2,CS3b,BC1,CS2a,CS2d,BC3,CS3d,BC5
BC2,CS3b,BC1,CS2a,CS2d,CS3d,BC3,BC5
BC2,CS3b,BC1,CS2a,CS2d,CS3d,BC5,BC3
BC2,CS3b,BC1,BC3,CS2a,CS2d,CS3d,BC5
BC2,CS3b,BC1,BC3,CS2d,CS2a,CS3d,BC5
BC2,CS3b,BC1,BC3,CS2d,CS3d,CS2a,BC5
BC2,CS3b,BC1,BC3,CS2d,CS3d,BC5,CS2a
BC2,CS3b,BC1,CS2d,CS2a,BC3,CS3d,BC5
BC2,CS3b,BC1,CS2d,CS2a,CS3d,BC3,BC5
BC2,CS3b,BC1,CS2d,CS2a,CS3d,BC5,BC3
BC2,CS3b,BC1,CS2d,BC3,CS2a,CS3d,BC5
BC2,CS3b,BC1,CS2d,BC3,CS3d,CS2a,BC5
BC2,CS3b,BC1,CS2d,BC3,CS3d,BC5,CS2a
BC2,CS3b,BC1,CS2d,CS3d,CS2a,BC3,BC5
BC2,CS3b,BC1,CS2d,CS3d,CS2a,BC5,BC3
BC2,CS3b,BC1,CS2d,CS3d,BC3,CS2a,BC5
BC2,CS3b,BC1,CS2d,CS3d,BC3,BC5,CS2a
BC2,CS3b,BC1,CS2d,CS3d,BC5,CS2a,BC3
BC2,CS3b,BC1,CS2d,CS3d,BC5,BC3,CS2a
BC2,CS3b,BC3,BC1,CS2a,CS2d,CS3d,BC5
BC2,CS3b,BC3,BC1,CS2d,CS2a,CS3d,BC5
BC2,CS3b,BC3,BC1,CS2d,CS3d,CS2a,BC5
BC2,CS3b,BC3,BC1,CS2d,CS3d,BC5,CS2a
BC2,CS3b,BC3,CS2d,BC1,CS2a,CS3d,BC5
BC2,CS3b,BC3,CS2d,BC1,CS3d,CS2a,BC5

BC2,CS3b,BC3,CS2d,BC1,CS3d,BC5,CS2a
BC2,CS3b,BC3,CS2d,CS3d,BC1,CS2a,BC5
BC2,CS3b,BC3,CS2d,CS3d,BC1,BC5,CS2a
BC2,CS3b,BC3,CS2d,CS3d,BC5,BC1,CS2a
BC2,CS3b,CS2d,BC1,CS2a,BC3,CS3d,BC5
BC2,CS3b,CS2d,BC1,CS2a,CS3d,BC3,BC5
BC2,CS3b,CS2d,BC1,CS2a,CS3d,BC5,BC3
BC2,CS3b,CS2d,BC1,BC3,CS2a,CS3d,BC5
BC2,CS3b,CS2d,BC1,BC3,CS3d,CS2a,BC5
BC2,CS3b,CS2d,BC1,BC3,CS3d,BC5,CS2a
BC2,CS3b,CS2d,BC1,CS3d,CS2a,BC3,BC5
BC2,CS3b,CS2d,BC1,CS3d,CS2a,BC5,BC3
BC2,CS3b,CS2d,BC1,CS3d,BC3,CS2a,BC5
BC2,CS3b,CS2d,BC1,CS3d,BC3,BC5,CS2a
BC2,CS3b,CS2d,BC1,CS3d,BC5,CS2a,BC3
BC2,CS3b,CS2d,BC1,CS3d,BC5,BC3,CS2a
BC2,CS3b,CS2d,BC3,BC1,CS2a,CS3d,BC5
BC2,CS3b,CS2d,BC3,BC1,CS3d,CS2a,BC5
BC2,CS3b,CS2d,BC3,BC1,CS3d,BC5,CS2a
BC2,CS3b,CS2d,BC3,CS3d,BC1,CS2a,BC5
BC2,CS3b,CS2d,BC3,CS3d,BC1,BC5,CS2a
BC2,CS3b,CS2d,BC3,CS3d,BC5,BC1,CS2a
BC2,CS3b,CS2d,CS3d,BC1,CS2a,BC3,BC5
BC2,CS3b,CS2d,CS3d,BC1,CS2a,BC5,BC3
BC2,CS3b,CS2d,CS3d,BC1,BC3,CS2a,BC5
BC2,CS3b,CS2d,CS3d,BC1,BC3,BC5,CS2a
BC2,CS3b,CS2d,CS3d,BC1,BC5,CS2a,BC3
BC2,CS3b,CS2d,CS3d,BC1,BC5,BC3,CS2a
BC2,CS3b,CS2d,CS3d,BC3,BC1,CS2a,BC5
BC2,CS3b,CS2d,CS3d,BC3,BC1,BC5,CS2a
BC2,CS3b,CS2d,CS3d,BC3,BC5,BC1,CS2a
BC2,CS3b,CS2d,CS3d,BC5,BC1,CS2a,BC3
BC2,CS3b,CS2d,CS3d,BC5,BC1,BC3,CS2a
BC2,CS3b,CS2d,CS3d,BC5,BC3,BC1,CS2a
BC2,CS2d,BC1,CS2a,CS3b,BC3,CS3d,BC5
BC2,CS2d,BC1,CS2a,CS3b,CS3d,BC3,BC5
BC2,CS2d,BC1,CS2a,CS3b,CS3d,BC5,BC3
BC2,CS2d,BC1,CS2a,CS3d,CS3b,BC3,BC5
BC2,CS2d,BC1,CS2a,CS3d,CS3b,BC5,BC3
BC2,CS2d,BC1,CS2a,CS3d,BC5,CS3b,BC3
BC2,CS2d,BC1,CS3b,CS2a,BC3,CS3d,BC5
BC2,CS2d,BC1,CS3b,CS2a,CS3d,BC3,BC5
BC2,CS2d,BC1,CS3b,CS2a,CS3d,BC5,BC3
BC2,CS2d,BC1,CS3b,BC3,CS2a,CS3d,BC5
BC2,CS2d,BC1,CS3b,BC3,CS3d,CS2a,BC5
BC2,CS2d,BC1,CS3b,BC3,CS3d,BC5,CS2a

BC2,CS2d,BC1,CS3b,CS3d,CS2a,BC3,BC5
BC2,CS2d,BC1,CS3b,CS3d,CS2a,BC5,BC3
BC2,CS2d,BC1,CS3b,CS3d,BC3,CS2a,BC5
BC2,CS2d,BC1,CS3b,CS3d,BC3,BC5,CS2a
BC2,CS2d,BC1,CS3b,CS3d,BC5,CS2a,BC3
BC2,CS2d,BC1,CS3b,CS3d,BC5,BC3,CS2a
BC2,CS2d,BC1,CS3d,CS2a,CS3b,BC3,BC5
BC2,CS2d,BC1,CS3d,CS2a,CS3b,BC5,BC3
BC2,CS2d,BC1,CS3d,CS2a,BC5,CS3b,BC3
BC2,CS2d,BC1,CS3d,CS3b,CS2a,BC3,BC5
BC2,CS2d,BC1,CS3d,CS3b,CS2a,BC5,BC3
BC2,CS2d,BC1,CS3d,CS3b,BC3,CS2a,BC5
BC2,CS2d,BC1,CS3d,CS3b,BC3,BC5,CS2a
BC2,CS2d,BC1,CS3d,CS3b,BC5,CS2a,BC3
BC2,CS2d,BC1,CS3d,CS3b,BC5,BC3,CS2a
BC2,CS2d,BC1,CS3d,BC5,CS2a,CS3b,BC3
BC2,CS2d,BC1,CS3d,BC5,CS3b,CS2a,BC3
BC2,CS2d,BC1,CS3d,BC5,CS3b,BC3,CS2a
BC2,CS2d,CS3b,BC1,CS2a,BC3,CS3d,BC5
BC2,CS2d,CS3b,BC1,CS2a,CS3d,BC3,BC5
BC2,CS2d,CS3b,BC1,CS2a,CS3d,BC5,BC3
BC2,CS2d,CS3b,BC1,BC3,CS2a,CS3d,BC5
BC2,CS2d,CS3b,BC1,BC3,CS3d,CS2a,BC5
BC2,CS2d,CS3b,BC1,BC3,CS3d,BC5,CS2a
BC2,CS2d,CS3b,BC1,CS3d,CS2a,BC3,BC5
BC2,CS2d,CS3b,BC1,CS3d,CS2a,BC5,BC3
BC2,CS2d,CS3b,BC1,CS3d,BC3,CS2a,BC5
BC2,CS2d,CS3b,BC1,CS3d,BC3,BC5,CS2a
BC2,CS2d,CS3b,BC1,CS3d,BC5,CS2a,BC3
BC2,CS2d,CS3b,BC1,CS3d,BC5,BC3,CS2a
BC2,CS2d,CS3b,BC3,BC1,CS2a,CS3d,BC5
BC2,CS2d,CS3b,BC3,BC1,CS3d,CS2a,BC5
BC2,CS2d,CS3b,BC3,BC1,CS3d,BC5,CS2a
BC2,CS2d,CS3b,BC3,CS3d,BC1,CS2a,BC5
BC2,CS2d,CS3b,BC3,CS3d,BC1,BC5,CS2a
BC2,CS2d,CS3b,BC3,CS3d,BC5,BC1,CS2a
BC2,CS2d,CS3b,CS3d,BC1,CS2a,BC3,BC5
BC2,CS2d,CS3b,CS3d,BC1,CS2a,BC5,BC3
BC2,CS2d,CS3b,CS3d,BC1,BC3,CS2a,BC5
BC2,CS2d,CS3b,CS3d,BC1,BC3,BC5,CS2a
BC2,CS2d,CS3b,CS3d,BC1,BC5,CS2a,BC3
BC2,CS2d,CS3b,CS3d,BC1,BC5,BC3,CS2a
BC2,CS2d,CS3b,CS3d,BC3,BC1,CS2a,BC5
BC2,CS2d,CS3b,CS3d,BC3,BC1,BC5,CS2a
BC2,CS2d,CS3b,CS3d,BC3,BC5,BC1,CS2a
BC2,CS2d,CS3b,CS3d,BC5,BC1,CS2a,BC3

BC2,CS2d,CS3b,CS3d,BC5,BC1,BC3,CS2a
BC2,CS2d,CS3b,CS3d,BC5,BC3,BC1,CS2a
BC2,CS2d,CS3d,BC1,CS2a,CS3b,BC3,BC5
BC2,CS2d,CS3d,BC1,CS2a,CS3b,BC5,BC3
BC2,CS2d,CS3d,BC1,CS2a,BC5,CS3b,BC3
BC2,CS2d,CS3d,BC1,CS3b,CS2a,BC3,BC5
BC2,CS2d,CS3d,BC1,CS3b,CS2a,BC5,BC3
BC2,CS2d,CS3d,BC1,CS3b,BC3,CS2a,BC5
BC2,CS2d,CS3d,BC1,CS3b,BC3,BC5,CS2a
BC2,CS2d,CS3d,BC1,CS3b,BC5,CS2a,BC3
BC2,CS2d,CS3d,BC1,CS3b,BC5,BC3,CS2a
BC2,CS2d,CS3d,BC1,BC5,CS2a,CS3b,BC3
BC2,CS2d,CS3d,BC1,BC5,CS3b,CS2a,BC3
BC2,CS2d,CS3d,BC1,BC5,CS3b,BC3,CS2a
BC2,CS2d,CS3d,CS3b,BC1,CS2a,BC3,BC5
BC2,CS2d,CS3d,CS3b,BC1,CS2a,BC5,BC3
BC2,CS2d,CS3d,CS3b,BC1,BC3,CS2a,BC5
BC2,CS2d,CS3d,CS3b,BC1,BC3,BC5,CS2a
BC2,CS2d,CS3d,CS3b,BC1,BC5,CS2a,BC3
BC2,CS2d,CS3d,CS3b,BC1,BC5,BC3,CS2a
BC2,CS2d,CS3d,CS3b,BC3,BC1,CS2a,BC5
BC2,CS2d,CS3d,CS3b,BC3,BC1,BC5,CS2a
BC2,CS2d,CS3d,CS3b,BC3,BC5,BC1,CS2a
BC2,CS2d,CS3d,CS3b,BC5,BC1,CS2a,BC3
BC2,CS2d,CS3d,CS3b,BC5,BC1,BC3,CS2a
BC2,CS2d,CS3d,CS3b,BC5,BC3,BC1,CS2a
BC2,CS2d,CS3d,BC5,BC1,CS2a,CS3b,BC3
BC2,CS2d,CS3d,BC5,BC1,CS3b,CS2a,BC3
BC2,CS2d,CS3d,BC5,BC1,CS3b,BC3,CS2a
BC2,CS2d,CS3d,BC5,CS3b,BC1,CS2a,BC3
BC2,CS2d,CS3d,BC5,CS3b,BC1,BC3,CS2a
BC2,CS2d,CS3d,BC5,CS3b,BC3,BC1,CS2a
CS2d,BC1,CS2a,BC2,CS3b,BC3,CS3d,BC5
CS2d,BC1,CS2a,BC2,CS3b,CS3d,BC3,BC5
CS2d,BC1,CS2a,BC2,CS3b,CS3d,BC5,BC3
CS2d,BC1,CS2a,BC2,CS3d,CS3b,BC3,BC5
CS2d,BC1,CS2a,BC2,CS3d,CS3b,BC5,BC3
CS2d,BC1,CS2a,BC2,CS3d,BC5,CS3b,BC3
CS2d,BC1,CS2a,CS3d,BC2,CS3b,BC3,BC5
CS2d,BC1,CS2a,CS3d,BC2,CS3b,BC5,BC3
CS2d,BC1,CS2a,CS3d,BC2,BC5,CS3b,BC3
CS2d,BC1,CS2a,CS3d,BC5,BC2,CS3b,BC3
CS2d,BC1,BC2,CS2a,CS3b,BC3,CS3d,BC5
CS2d,BC1,BC2,CS2a,CS3b,CS3d,BC3,BC5
CS2d,BC1,BC2,CS2a,CS3b,CS3d,BC5,BC3
CS2d,BC1,BC2,CS2a,CS3d,CS3b,BC3,BC5

CS2d,BC1,BC2,CS2a,CS3d,CS3b,BC5,BC3
CS2d,BC1,BC2,CS2a,CS3d,BC5,CS3b,BC3
CS2d,BC1,BC2,CS3b,CS2a,BC3,CS3d,BC5
CS2d,BC1,BC2,CS3b,CS2a,CS3d,BC3,BC5
CS2d,BC1,BC2,CS3b,CS2a,CS3d,BC5,BC3
CS2d,BC1,BC2,CS3b,BC3,CS2a,CS3d,BC5
CS2d,BC1,BC2,CS3b,BC3,CS3d,CS2a,BC5
CS2d,BC1,BC2,CS3b,BC3,CS3d,BC5,CS2a
CS2d,BC1,BC2,CS3b,CS3d,CS2a,BC3,BC5
CS2d,BC1,BC2,CS3b,CS3d,CS2a,BC5,BC3
CS2d,BC1,BC2,CS3b,CS3d,BC3,CS2a,BC5
CS2d,BC1,BC2,CS3b,CS3d,BC3,BC5,CS2a
CS2d,BC1,BC2,CS3b,CS3d,BC5,CS2a,BC3
CS2d,BC1,BC2,CS3b,CS3d,BC5,BC3,CS2a
CS2d,BC1,BC2,CS3d,CS2a,CS3b,BC3,BC5
CS2d,BC1,BC2,CS3d,CS2a,CS3b,BC5,BC3
CS2d,BC1,BC2,CS3d,CS2a,BC5,CS3b,BC3
CS2d,BC1,BC2,CS3d,CS3b,CS2a,BC3,BC5
CS2d,BC1,BC2,CS3d,CS3b,CS2a,BC5,BC3
CS2d,BC1,BC2,CS3d,CS3b,BC3,CS2a,BC5
CS2d,BC1,BC2,CS3d,CS3b,BC3,BC5,CS2a
CS2d,BC1,BC2,CS3d,CS3b,BC5,CS2a,BC3
CS2d,BC1,BC2,CS3d,CS3b,BC5,BC3,CS2a
CS2d,BC1,BC2,CS3d,BC5,CS2a,CS3b,BC3
CS2d,BC1,BC2,CS3d,BC5,CS3b,CS2a,BC3
CS2d,BC1,BC2,CS3d,BC5,CS3b,BC3,CS2a
CS2d,BC1,CS3d,CS2a,BC2,CS3b,BC3,BC5
CS2d,BC1,CS3d,CS2a,BC2,CS3b,BC5,BC3
CS2d,BC1,CS3d,CS2a,BC2,BC5,CS3b,BC3
CS2d,BC1,CS3d,CS2a,BC5,BC2,CS3b,BC3
CS2d,BC1,CS3d,BC2,CS2a,CS3b,BC3,BC5
CS2d,BC1,CS3d,BC2,CS2a,CS3b,BC5,BC3
CS2d,BC1,CS3d,BC2,CS2a,BC5,CS3b,BC3
CS2d,BC1,CS3d,BC2,CS3b,CS2a,BC3,BC5
CS2d,BC1,CS3d,BC2,CS3b,CS2a,BC5,BC3
CS2d,BC1,CS3d,BC2,CS3b,BC3,CS2a,BC5
CS2d,BC1,CS3d,BC2,CS3b,BC3,BC5,CS2a
CS2d,BC1,CS3d,BC2,CS3b,BC5,CS2a,BC3
CS2d,BC1,CS3d,BC2,CS3b,BC5,BC3,CS2a
CS2d,BC1,CS3d,BC2,BC5,CS2a,CS3b,BC3
CS2d,BC1,CS3d,BC2,BC5,CS3b,CS2a,BC3
CS2d,BC1,CS3d,BC2,BC5,CS3b,BC3,CS2a
CS2d,BC1,CS3d,BC5,CS2a,BC2,CS3b,BC3
CS2d,BC1,CS3d,BC5,BC2,CS2a,CS3b,BC3
CS2d,BC1,CS3d,BC5,BC2,CS3b,CS2a,BC3
CS2d,BC1,CS3d,BC5,BC2,CS3b,BC3,CS2a

CS2d,BC2,BC1,CS2a,CS3b,BC3,CS3d,BC5
CS2d,BC2,BC1,CS2a,CS3b,CS3d,BC3,BC5
CS2d,BC2,BC1,CS2a,CS3b,CS3d,BC5,BC3
CS2d,BC2,BC1,CS2a,CS3d,CS3b,BC3,BC5
CS2d,BC2,BC1,CS2a,CS3d,CS3b,BC5,BC3
CS2d,BC2,BC1,CS2a,CS3d,BC5,CS3b,BC3
CS2d,BC2,BC1,CS3b,CS2a,BC3,CS3d,BC5
CS2d,BC2,BC1,CS3b,CS2a,CS3d,BC3,BC5
CS2d,BC2,BC1,CS3b,CS2a,CS3d,BC5,BC3
CS2d,BC2,BC1,CS3b,BC3,CS2a,CS3d,BC5
CS2d,BC2,BC1,CS3b,BC3,CS3d,CS2a,BC5
CS2d,BC2,BC1,CS3b,BC3,CS3d,BC5,CS2a
CS2d,BC2,BC1,CS3b,CS3d,CS2a,BC3,BC5
CS2d,BC2,BC1,CS3b,CS3d,CS2a,BC5,BC3
CS2d,BC2,BC1,CS3b,CS3d,BC3,CS2a,BC5
CS2d,BC2,BC1,CS3b,CS3d,BC3,BC5,CS2a
CS2d,BC2,BC1,CS3b,CS3d,BC5,CS2a,BC3
CS2d,BC2,BC1,CS3b,CS3d,BC5,BC3,CS2a
CS2d,BC2,BC1,CS3d,CS2a,CS3b,BC3,BC5
CS2d,BC2,BC1,CS3d,CS2a,CS3b,BC5,BC3
CS2d,BC2,BC1,CS3d,CS2a,BC5,CS3b,BC3
CS2d,BC2,BC1,CS3d,CS3b,CS2a,BC3,BC5
CS2d,BC2,BC1,CS3d,CS3b,CS2a,BC5,BC3
CS2d,BC2,BC1,CS3d,CS3b,BC3,CS2a,BC5
CS2d,BC2,BC1,CS3d,CS3b,BC3,BC5,CS2a
CS2d,BC2,BC1,CS3d,CS3b,BC5,CS2a,BC3
CS2d,BC2,BC1,CS3d,CS3b,BC5,BC3,CS2a
CS2d,BC2,BC1,CS3d,BC5,CS2a,CS3b,BC3
CS2d,BC2,BC1,CS3d,BC5,CS3b,CS2a,BC3
CS2d,BC2,BC1,CS3d,BC5,CS3b,BC3,CS2a
CS2d,BC2,CS3b,BC1,CS2a,BC3,CS3d,BC5
CS2d,BC2,CS3b,BC1,CS2a,CS3d,BC3,BC5
CS2d,BC2,CS3b,BC1,CS2a,CS3d,BC5,BC3
CS2d,BC2,CS3b,BC1,BC3,CS2a,CS3d,BC5
CS2d,BC2,CS3b,BC1,BC3,CS3d,CS2a,BC5
CS2d,BC2,CS3b,BC1,BC3,CS3d,BC5,CS2a
CS2d,BC2,CS3b,BC1,CS3d,CS2a,BC3,BC5
CS2d,BC2,CS3b,BC1,CS3d,CS2a,BC5,BC3
CS2d,BC2,CS3b,BC1,CS3d,BC3,CS2a,BC5
CS2d,BC2,CS3b,BC1,CS3d,BC3,BC5,CS2a
CS2d,BC2,CS3b,BC1,CS3d,BC5,CS2a,BC3
CS2d,BC2,CS3b,BC1,CS3d,BC5,BC3,CS2a
CS2d,BC2,CS3b,BC3,BC1,CS2a,CS3d,BC5
CS2d,BC2,CS3b,BC3,BC1,CS3d,CS2a,BC5
CS2d,BC2,CS3b,BC3,BC1,CS3d,BC5,CS2a
CS2d,BC2,CS3b,BC3,CS3d,BC1,CS2a,BC5

CS2d,BC2,CS3b,BC3,CS3d,BC1,BC5,CS2a
CS2d,BC2,CS3b,BC3,CS3d,BC5,BC1,CS2a
CS2d,BC2,CS3b,CS3d,BC1,CS2a,BC3,BC5
CS2d,BC2,CS3b,CS3d,BC1,CS2a,BC5,BC3
CS2d,BC2,CS3b,CS3d,BC1,BC3,CS2a,BC5
CS2d,BC2,CS3b,CS3d,BC1,BC3,BC5,CS2a
CS2d,BC2,CS3b,CS3d,BC1,BC5,CS2a,BC3
CS2d,BC2,CS3b,CS3d,BC1,BC5,BC3,CS2a
CS2d,BC2,CS3b,CS3d,BC3,BC1,CS2a,BC5
CS2d,BC2,CS3b,CS3d,BC3,BC1,BC5,CS2a
CS2d,BC2,CS3b,CS3d,BC3,BC5,BC1,CS2a
CS2d,BC2,CS3b,CS3d,BC5,BC1,CS2a,BC3
CS2d,BC2,CS3b,CS3d,BC5,BC1,BC3,CS2a
CS2d,BC2,CS3b,CS3d,BC5,BC3,BC1,CS2a
CS2d,BC2,CS3d,BC1,CS2a,CS3b,BC3,BC5
CS2d,BC2,CS3d,BC1,CS2a,CS3b,BC5,BC3
CS2d,BC2,CS3d,BC1,CS2a,BC5,CS3b,BC3
CS2d,BC2,CS3d,BC1,CS3b,CS2a,BC3,BC5
CS2d,BC2,CS3d,BC1,CS3b,CS2a,BC5,BC3
CS2d,BC2,CS3d,BC1,CS3b,BC3,CS2a,BC5
CS2d,BC2,CS3d,BC1,CS3b,BC3,BC5,CS2a
CS2d,BC2,CS3d,BC1,CS3b,BC5,CS2a,BC3
CS2d,BC2,CS3d,BC1,CS3b,BC5,BC3,CS2a
CS2d,BC2,CS3d,BC1,BC5,CS2a,CS3b,BC3
CS2d,BC2,CS3d,BC1,BC5,CS3b,CS2a,BC3
CS2d,BC2,CS3d,BC1,BC5,CS3b,BC3,CS2a
CS2d,BC2,CS3d,CS3b,BC1,CS2a,BC3,BC5
CS2d,BC2,CS3d,CS3b,BC1,CS2a,BC5,BC3
CS2d,BC2,CS3d,CS3b,BC1,BC3,CS2a,BC5
CS2d,BC2,CS3d,CS3b,BC1,BC3,BC5,CS2a
CS2d,BC2,CS3d,CS3b,BC1,BC5,CS2a,BC3
CS2d,BC2,CS3d,CS3b,BC1,BC5,BC3,CS2a
CS2d,BC2,CS3d,CS3b,BC3,BC1,CS2a,BC5
CS2d,BC2,CS3d,CS3b,BC3,BC1,BC5,CS2a
CS2d,BC2,CS3d,CS3b,BC3,BC5,BC1,CS2a
CS2d,BC2,CS3d,CS3b,BC5,BC1,CS2a,BC3
CS2d,BC2,CS3d,CS3b,BC5,BC1,BC3,CS2a
CS2d,BC2,CS3d,CS3b,BC5,BC3,BC1,CS2a
CS2d,BC2,CS3d,BC5,BC1,CS2a,CS3b,BC3
CS2d,BC2,CS3d,BC5,BC1,CS3b,CS2a,BC3
CS2d,BC2,CS3d,BC5,BC1,CS3b,BC3,CS2a
CS2d,BC2,CS3d,BC5,CS3b,BC1,CS2a,BC3
CS2d,BC2,CS3d,BC5,CS3b,BC1,BC3,CS2a
CS2d,BC2,CS3d,BC5,CS3b,BC3,BC1,CS2a
CS2d,CS3d,BC1,CS2a,BC2,CS3b,BC3,BC5
CS2d,CS3d,BC1,CS2a,BC2,CS3b,BC5,BC3

CS2d,CS3d,BC1,CS2a,BC2,BC5,CS3b,BC3
CS2d,CS3d,BC1,CS2a,BC5,BC2,CS3b,BC3
CS2d,CS3d,BC1,BC2,CS2a,CS3b,BC3,BC5
CS2d,CS3d,BC1,BC2,CS2a,CS3b,BC5,BC3
CS2d,CS3d,BC1,BC2,CS2a,BC5,CS3b,BC3
CS2d,CS3d,BC1,BC2,CS3b,CS2a,BC3,BC5
CS2d,CS3d,BC1,BC2,CS3b,CS2a,BC5,BC3
CS2d,CS3d,BC1,BC2,CS3b,BC3,CS2a,BC5
CS2d,CS3d,BC1,BC2,CS3b,BC3,BC5,CS2a
CS2d,CS3d,BC1,BC2,CS3b,BC5,CS2a,BC3
CS2d,CS3d,BC1,BC2,CS3b,BC5,BC3,CS2a
CS2d,CS3d,BC1,BC2,BC5,CS2a,CS3b,BC3
CS2d,CS3d,BC1,BC2,BC5,CS3b,CS2a,BC3
CS2d,CS3d,BC1,BC2,BC5,CS3b,BC3,CS2a
CS2d,CS3d,BC1,BC5,CS2a,BC2,CS3b,BC3
CS2d,CS3d,BC1,BC5,BC2,CS2a,CS3b,BC3
CS2d,CS3d,BC1,BC5,BC2,CS3b,CS2a,BC3
CS2d,CS3d,BC1,BC5,BC2,CS3b,BC3,CS2a
CS2d,CS3d,BC2,BC1,CS2a,CS3b,BC3,BC5
CS2d,CS3d,BC2,BC1,CS2a,CS3b,BC5,BC3
CS2d,CS3d,BC2,BC1,CS2a,BC5,CS3b,BC3
CS2d,CS3d,BC2,BC1,CS3b,CS2a,BC3,BC5
CS2d,CS3d,BC2,BC1,CS3b,CS2a,BC5,BC3
CS2d,CS3d,BC2,BC1,CS3b,BC3,CS2a,BC5
CS2d,CS3d,BC2,BC1,CS3b,BC3,BC5,CS2a
CS2d,CS3d,BC2,BC1,CS3b,BC5,CS2a,BC3
CS2d,CS3d,BC2,BC1,CS3b,BC5,BC3,CS2a
CS2d,CS3d,BC2,BC1,BC5,CS2a,CS3b,BC3
CS2d,CS3d,BC2,BC1,BC5,CS3b,CS2a,BC3
CS2d,CS3d,BC2,BC1,BC5,CS3b,BC3,CS2a
CS2d,CS3d,BC2,CS3b,BC1,CS2a,BC3,BC5
CS2d,CS3d,BC2,CS3b,BC1,CS2a,BC5,BC3
CS2d,CS3d,BC2,CS3b,BC1,BC3,CS2a,BC5
CS2d,CS3d,BC2,CS3b,BC1,BC3,BC5,CS2a
CS2d,CS3d,BC2,CS3b,BC1,BC5,CS2a,BC3
CS2d,CS3d,BC2,CS3b,BC1,BC5,BC3,CS2a
CS2d,CS3d,BC2,CS3b,BC3,BC1,CS2a,BC5
CS2d,CS3d,BC2,CS3b,BC3,BC1,BC5,CS2a
CS2d,CS3d,BC2,CS3b,BC3,BC5,BC1,CS2a
CS2d,CS3d,BC2,CS3b,BC5,BC1,CS2a,BC3
CS2d,CS3d,BC2,CS3b,BC5,BC1,BC3,CS2a
CS2d,CS3d,BC2,CS3b,BC5,BC3,BC1,CS2a
CS2d,CS3d,BC2,BC5,BC1,CS2a,CS3b,BC3
CS2d,CS3d,BC2,BC5,BC1,CS3b,CS2a,BC3
CS2d,CS3d,BC2,BC5,BC1,CS3b,BC3,CS2a
CS2d,CS3d,BC2,BC5,CS3b,BC1,CS2a,BC3

CS2d,CS3d,BC2,BC5,CS3b,BC1,BC3,CS2a
CS2d,CS3d,BC2,BC5,CS3b,BC3,BC1,CS2a
CS2d,CS3d,BC5,BC1,CS2a,BC2,CS3b,BC3
CS2d,CS3d,BC5,BC1,BC2,CS2a,CS3b,BC3
CS2d,CS3d,BC5,BC1,BC2,CS3b,CS2a,BC3
CS2d,CS3d,BC5,BC1,BC2,CS3b,BC3,CS2a
CS2d,CS3d,BC5,BC2,BC1,CS2a,CS3b,BC3
CS2d,CS3d,BC5,BC2,BC1,CS3b,CS2a,BC3
CS2d,CS3d,BC5,BC2,BC1,CS3b,BC3,CS2a
CS2d,CS3d,BC5,BC2,CS3b,BC1,CS2a,BC3
CS2d,CS3d,BC5,BC2,CS3b,BC1,BC3,CS2a
CS2d,CS3d,BC5,BC2,CS3b,BC3,BC1,CS2a

Appendix H

Equivalent schedule classes for Case 4

BC1,CS2a,BC2,CS3b,BC3,CS2d,CS3d,BC5
BC1,CS2a,BC2,CS3b,CS2d,BC3,CS3d,BC5
BC1,CS2a,BC2,CS3b,CS2d,CS3d,BC3,BC5
BC1,CS2a,BC2,CS2d,CS3b,BC3,CS3d,BC5
BC1,CS2a,BC2,CS2d,CS3b,CS3d,BC3,BC5
BC1,CS2a,BC2,CS2d,CS3d,CS3b,BC3,BC5
BC1,CS2a,BC2,CS2d,CS3d,BC5,CS3b,BC3
BC1,CS2a,CS2d,BC2,CS3b,BC3,CS3d,BC5
BC1,CS2a,CS2d,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3d,CS3b,BC3,BC5
BC1,CS2a,CS2d,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2a,CS2d,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2a,CS2d,CS3d,BC2,BC5,CS3b,BC3
BC1,BC2,CS3b,BC3,CS2d,CS2a,CS3d,BC5
BC1,BC2,CS3b,BC3,CS2d,CS3d,CS2a,BC5
BC1,BC2,CS3b,BC3,CS2d,CS3d,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS2a,BC3,CS3d,BC5
BC1,BC2,CS3b,CS2d,CS2a,CS3d,BC3,BC5
BC1,BC2,CS3b,CS2d,BC3,CS2a,CS3d,BC5
BC1,BC2,CS3b,CS2d,BC3,CS3d,CS2a,BC5
BC1,BC2,CS3b,CS2d,BC3,CS3d,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS3d,CS2a,BC3,BC5
BC1,BC2,CS3b,CS2d,CS3d,BC3,CS2a,BC5
BC1,BC2,CS3b,CS2d,CS3d,BC3,BC5,CS2a
BC1,BC2,CS3b,CS2d,CS3d,BC5,CS2a,BC3
BC1,BC2,CS3b,CS2d,CS3d,BC5,BC3,CS2a
BC1,BC2,CS2d,CS2a,CS3b,BC3,CS3d,BC5
BC1,BC2,CS2d,CS2a,CS3b,CS3d,BC3,BC5
BC1,BC2,CS2d,CS2a,CS3d,CS3b,BC3,BC5
BC1,BC2,CS2d,CS2a,CS3d,BC5,CS3b,BC3
BC1,BC2,CS2d,CS3d,CS2a,CS3b,BC3,BC5
BC1,BC2,CS2d,CS3d,CS2a,BC5,CS3b,BC3
BC1,BC2,CS2d,CS3d,CS3b,CS2a,BC3,BC5
BC1,BC2,CS2d,CS3d,CS3b,BC3,CS2a,BC5
BC1,BC2,CS2d,CS3d,CS3b,BC3,BC5,CS2a
BC1,BC2,CS2d,CS3d,CS3b,BC5,CS2a,BC3
BC1,BC2,CS2d,CS3d,BC5,CS2a,CS3b,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,CS2a,BC3
BC1,BC2,CS2d,CS3d,BC5,CS3b,BC3,CS2a
BC1,CS2d,CS2a,BC2,CS3b,BC3,CS3d,BC5
BC1,CS2d,CS2a,BC2,CS3b,CS3d,BC3,BC5
BC1,CS2d,CS2a,BC2,CS3d,CS3b,BC3,BC5

BC1,CS2d,CS2a,BC2,CS3d,BC5,CS3b,BC3
BC1,CS2d,CS2a,CS3d,BC2,CS3b,BC3,BC5
BC1,CS2d,CS2a,CS3d,BC2,BC5,CS3b,BC3
BC1,CS2d,CS3d,CS2a,BC2,CS3b,BC3,BC5
BC1,CS2d,CS3d,CS2a,BC2,BC5,CS3b,BC3
BC1,CS2d,CS3d,BC5,CS2a,BC2,CS3b,BC3
BC2,CS3b,BC3,CS2d,CS3d,BC1,CS2a,BC5
BC2,CS3b,BC3,CS2d,CS3d,BC1,BC5,CS2a
BC2,CS3b,CS2d,CS3d,BC1,CS2a,BC3,BC5
BC2,CS3b,CS2d,CS3d,BC1,BC3,CS2a,BC5
BC2,CS3b,CS2d,CS3d,BC1,BC3,BC5,CS2a
BC2,CS3b,CS2d,CS3d,BC1,BC5,CS2a,BC3
BC2,CS2d,CS3d,CS3b,BC1,CS2a,BC3,BC5
BC2,CS2d,CS3d,CS3b,BC1,BC3,CS2a,BC5
BC2,CS2d,CS3d,CS3b,BC1,BC3,BC5,CS2a
BC2,CS2d,CS3d,CS3b,BC1,BC5,CS2a,BC3
BC2,CS2d,CS3d,BC5,CS3b,BC1,CS2a,BC3
BC2,CS2d,CS3d,BC5,CS3b,BC1,BC3,CS2a

Appendix I

All legal schedules for Case 5.

BC1,CS1a,BC2,CS1b,BC3,CS1c,BC4,CS1d
BC1,CS1a,BC2,CS1b,BC3,BC4,CS1c,CS1d
BC1,CS1a,BC2,CS1b,BC3,BC4,CS1d,CS1c
BC1,CS1a,BC2,CS1b,BC4,BC3,CS1c,CS1d
BC1,CS1a,BC2,CS1b,BC4,BC3,CS1d,CS1c
BC1,CS1a,BC2,CS1b,BC4,CS1d,BC3,CS1c
BC1,CS1a,BC2,BC3,CS1b,CS1c,BC4,CS1d
BC1,CS1a,BC2,BC3,CS1b,BC4,CS1c,CS1d
BC1,CS1a,BC2,BC3,CS1b,BC4,CS1d,CS1c
BC1,CS1a,BC2,BC3,CS1c,CS1b,BC4,CS1d
BC1,CS1a,BC2,BC3,CS1c,BC4,CS1b,CS1d
BC1,CS1a,BC2,BC3,CS1c,BC4,CS1d,CS1b
BC1,CS1a,BC2,BC3,BC4,CS1b,CS1c,CS1d
BC1,CS1a,BC2,BC3,BC4,CS1b,CS1d,CS1c
BC1,CS1a,BC2,BC3,BC4,CS1c,CS1b,CS1d
BC1,CS1a,BC2,BC3,BC4,CS1c,CS1d,CS1b
BC1,CS1a,BC2,BC3,BC4,CS1d,CS1b,CS1c
BC1,CS1a,BC2,BC3,BC4,CS1d,CS1c,CS1b
BC1,CS1a,BC2,BC4,CS1b,BC3,CS1c,CS1d
BC1,CS1a,BC2,BC4,CS1b,BC3,CS1d,CS1c
BC1,CS1a,BC2,BC4,CS1b,CS1d,BC3,CS1c
BC1,CS1a,BC2,BC4,BC3,CS1b,CS1c,CS1d
BC1,CS1a,BC2,BC4,BC3,CS1b,CS1d,CS1c
BC1,CS1a,BC2,BC4,BC3,CS1c,CS1b,CS1d
BC1,CS1a,BC2,BC4,BC3,CS1c,CS1d,CS1b
BC1,CS1a,BC2,BC4,BC3,CS1d,CS1b,CS1c
BC1,CS1a,BC2,BC4,BC3,CS1d,CS1c,CS1b
BC1,CS1a,BC2,BC4,CS1d,CS1b,BC3,CS1c
BC1,CS1a,BC2,BC4,CS1d,BC3,CS1b,CS1c
BC1,CS1a,BC2,BC4,CS1d,BC3,CS1c,CS1b
BC1,CS1a,BC3,BC2,CS1b,CS1c,BC4,CS1d
BC1,CS1a,BC3,BC2,CS1b,BC4,CS1c,CS1d
BC1,CS1a,BC3,BC2,CS1b,BC4,CS1d,CS1c
BC1,CS1a,BC3,BC2,CS1c,CS1b,BC4,CS1d
BC1,CS1a,BC3,BC2,CS1c,BC4,CS1b,CS1d
BC1,CS1a,BC3,BC2,CS1c,BC4,CS1d,CS1b
BC1,CS1a,BC3,BC2,BC4,CS1b,CS1c,CS1d
BC1,CS1a,BC3,BC2,BC4,CS1b,CS1d,CS1c
BC1,CS1a,BC3,BC2,BC4,CS1c,CS1b,CS1d
BC1,CS1a,BC3,BC2,BC4,CS1c,CS1d,CS1b
BC1,CS1a,BC3,BC2,BC4,CS1d,CS1b,CS1c
BC1,CS1a,BC3,BC2,BC4,CS1d,CS1c,CS1b

BC1,CS1a,BC3,CS1c,BC2,CS1b,BC4,CS1d
BC1,CS1a,BC3,CS1c,BC2,BC4,CS1b,CS1d
BC1,CS1a,BC3,CS1c,BC2,BC4,CS1d,CS1b
BC1,CS1a,BC3,CS1c,BC4,BC2,CS1b,CS1d
BC1,CS1a,BC3,CS1c,BC4,BC2,CS1d,CS1b
BC1,CS1a,BC3,CS1c,BC4,CS1d,BC2,CS1b
BC1,CS1a,BC3,BC4,BC2,CS1b,CS1c,CS1d
BC1,CS1a,BC3,BC4,BC2,CS1b,CS1d,CS1c
BC1,CS1a,BC3,BC4,BC2,CS1c,CS1b,CS1d
BC1,CS1a,BC3,BC4,BC2,CS1c,CS1d,CS1b
BC1,CS1a,BC3,BC4,BC2,CS1d,CS1b,CS1c
BC1,CS1a,BC3,BC4,BC2,CS1d,CS1c,CS1b
BC1,CS1a,BC3,BC4,CS1c,BC2,CS1b,CS1d
BC1,CS1a,BC3,BC4,CS1c,BC2,CS1d,CS1b
BC1,CS1a,BC3,BC4,CS1c,CS1d,BC2,CS1b
BC1,CS1a,BC3,BC4,CS1d,BC2,CS1b,CS1c
BC1,CS1a,BC3,BC4,CS1d,BC2,CS1c,CS1b
BC1,CS1a,BC3,BC4,CS1d,CS1c,BC2,CS1b
BC1,CS1a,BC4,BC2,CS1b,BC3,CS1c,CS1d
BC1,CS1a,BC4,BC2,CS1b,BC3,CS1d,CS1c
BC1,CS1a,BC4,BC2,CS1b,CS1d,BC3,CS1c
BC1,CS1a,BC4,BC2,BC3,CS1b,CS1c,CS1d
BC1,CS1a,BC4,BC2,BC3,CS1b,CS1d,CS1c
BC1,CS1a,BC4,BC2,BC3,CS1c,CS1b,CS1d
BC1,CS1a,BC4,BC2,BC3,CS1c,CS1d,CS1b
BC1,CS1a,BC4,BC2,BC3,CS1d,CS1b,CS1c
BC1,CS1a,BC4,BC2,BC3,CS1d,CS1c,CS1b
BC1,CS1a,BC4,BC2,CS1d,CS1b,BC3,CS1c
BC1,CS1a,BC4,BC2,CS1d,BC3,CS1b,CS1c
BC1,CS1a,BC4,BC2,CS1d,BC3,CS1c,CS1b
BC1,CS1a,BC4,BC3,BC2,CS1b,CS1c,CS1d
BC1,CS1a,BC4,BC3,BC2,CS1b,CS1d,CS1c
BC1,CS1a,BC4,BC3,BC2,CS1c,CS1b,CS1d
BC1,CS1a,BC4,BC3,BC2,CS1c,CS1d,CS1b
BC1,CS1a,BC4,BC3,BC2,CS1d,CS1b,CS1c
BC1,CS1a,BC4,BC3,BC2,CS1d,CS1c,CS1b
BC1,CS1a,BC4,BC3,CS1c,BC2,CS1b,CS1d
BC1,CS1a,BC4,BC3,CS1c,BC2,CS1d,CS1b
BC1,CS1a,BC4,BC3,CS1c,CS1d,BC2,CS1b
BC1,CS1a,BC4,BC3,CS1d,BC2,CS1b,CS1c
BC1,CS1a,BC4,BC3,CS1d,BC2,CS1c,CS1b
BC1,CS1a,BC4,BC3,CS1d,CS1c,BC2,CS1b
BC1,CS1a,BC4,CS1d,BC2,CS1b,BC3,CS1c
BC1,CS1a,BC4,CS1d,BC2,BC3,CS1b,CS1c
BC1,CS1a,BC4,CS1d,BC2,BC3,CS1c,CS1b
BC1,CS1a,BC4,CS1d,BC3,BC2,CS1b,CS1c

BC1,CS1a,BC4,CS1d,BC3,BC2,CS1c,CS1b
BC1,CS1a,BC4,CS1d,BC3,CS1c,BC2,CS1b
BC1,BC2,CS1a,CS1b,BC3,CS1c,BC4,CS1d
BC1,BC2,CS1a,CS1b,BC3,BC4,CS1c,CS1d
BC1,BC2,CS1a,CS1b,BC3,BC4,CS1d,CS1c
BC1,BC2,CS1a,CS1b,BC4,BC3,CS1c,CS1d
BC1,BC2,CS1a,CS1b,BC4,BC3,CS1d,CS1c
BC1,BC2,CS1a,CS1b,BC4,CS1d,BC3,CS1c
BC1,BC2,CS1a,BC3,CS1b,CS1c,BC4,CS1d
BC1,BC2,CS1a,BC3,CS1b,BC4,CS1c,CS1d
BC1,BC2,CS1a,BC3,CS1b,BC4,CS1d,CS1c
BC1,BC2,CS1a,BC3,CS1c,CS1b,BC4,CS1d
BC1,BC2,CS1a,BC3,CS1c,BC4,CS1b,CS1d
BC1,BC2,CS1a,BC3,CS1c,BC4,CS1d,CS1b
BC1,BC2,CS1a,BC3,BC4,CS1b,CS1c,CS1d
BC1,BC2,CS1a,BC3,BC4,CS1b,CS1d,CS1c
BC1,BC2,CS1a,BC3,BC4,CS1c,CS1b,CS1d
BC1,BC2,CS1a,BC3,BC4,CS1c,CS1d,CS1b
BC1,BC2,CS1a,BC3,BC4,CS1d,CS1b,CS1c
BC1,BC2,CS1a,BC3,BC4,CS1d,CS1c,CS1b
BC1,BC2,CS1a,BC4,CS1b,BC3,CS1c,CS1d
BC1,BC2,CS1a,BC4,CS1b,BC3,CS1d,CS1c
BC1,BC2,CS1a,BC4,CS1b,CS1d,BC3,CS1c
BC1,BC2,CS1a,BC4,BC3,CS1b,CS1c,CS1d
BC1,BC2,CS1a,BC4,BC3,CS1b,CS1d,CS1c
BC1,BC2,CS1a,BC4,BC3,CS1c,CS1b,CS1d
BC1,BC2,CS1a,BC4,BC3,CS1c,CS1d,CS1b
BC1,BC2,CS1a,BC4,BC3,CS1d,CS1b,CS1c
BC1,BC2,CS1a,BC4,BC3,CS1d,CS1c,CS1b
BC1,BC2,CS1a,BC4,CS1d,CS1b,BC3,CS1c
BC1,BC2,CS1a,BC4,CS1d,BC3,CS1b,CS1c
BC1,BC2,CS1a,BC4,CS1d,BC3,CS1c,CS1b
BC1,BC2,CS1b,CS1a,BC3,CS1c,BC4,CS1d
BC1,BC2,CS1b,CS1a,BC3,BC4,CS1c,CS1d
BC1,BC2,CS1b,CS1a,BC3,BC4,CS1d,CS1c
BC1,BC2,CS1b,CS1a,BC4,BC3,CS1c,CS1d
BC1,BC2,CS1b,CS1a,BC4,BC3,CS1d,CS1c
BC1,BC2,CS1b,CS1a,BC4,CS1d,BC3,CS1c
BC1,BC2,CS1b,BC3,CS1a,CS1c,BC4,CS1d
BC1,BC2,CS1b,BC3,CS1a,BC4,CS1c,CS1d
BC1,BC2,CS1b,BC3,CS1a,BC4,CS1d,CS1c
BC1,BC2,CS1b,BC3,CS1c,CS1a,BC4,CS1d
BC1,BC2,CS1b,BC3,CS1c,BC4,CS1a,CS1d
BC1,BC2,CS1b,BC3,CS1c,BC4,CS1d,CS1a
BC1,BC2,CS1b,BC3,BC4,CS1a,CS1c,CS1d
BC1,BC2,CS1b,BC3,BC4,CS1a,CS1d,CS1c

BC1,BC2,CS1b,BC3,BC4,CS1c,CS1a,CS1d
BC1,BC2,CS1b,BC3,BC4,CS1c,CS1d,CS1a
BC1,BC2,CS1b,BC3,BC4,CS1d,CS1a,CS1c
BC1,BC2,CS1b,BC3,BC4,CS1d,CS1c,CS1a
BC1,BC2,CS1b,BC4,CS1a,BC3,CS1c,CS1d
BC1,BC2,CS1b,BC4,CS1a,BC3,CS1d,CS1c
BC1,BC2,CS1b,BC4,CS1a,CS1d,BC3,CS1c
BC1,BC2,CS1b,BC4,BC3,CS1a,CS1c,CS1d
BC1,BC2,CS1b,BC4,BC3,CS1a,CS1d,CS1c
BC1,BC2,CS1b,BC4,BC3,CS1c,CS1a,CS1d
BC1,BC2,CS1b,BC4,BC3,CS1c,CS1d,CS1a
BC1,BC2,CS1b,BC4,BC3,CS1d,CS1a,CS1c
BC1,BC2,CS1b,BC4,BC3,CS1d,CS1c,CS1a
BC1,BC2,CS1b,BC4,CS1d,CS1a,BC3,CS1c
BC1,BC2,CS1b,BC4,CS1d,BC3,CS1a,CS1c
BC1,BC2,CS1b,BC4,CS1d,BC3,CS1c,CS1a
BC1,BC2,BC3,CS1a,CS1b,CS1c,BC4,CS1d
BC1,BC2,BC3,CS1a,CS1b,BC4,CS1c,CS1d
BC1,BC2,BC3,CS1a,CS1b,BC4,CS1d,CS1c
BC1,BC2,BC3,CS1a,CS1c,CS1b,BC4,CS1d
BC1,BC2,BC3,CS1a,CS1c,BC4,CS1b,CS1d
BC1,BC2,BC3,CS1a,CS1c,BC4,CS1d,CS1b
BC1,BC2,BC3,CS1a,BC4,CS1b,CS1c,CS1d
BC1,BC2,BC3,CS1a,BC4,CS1b,CS1d,CS1c
BC1,BC2,BC3,CS1a,BC4,CS1c,CS1b,CS1d
BC1,BC2,BC3,CS1a,BC4,CS1c,CS1d,CS1b
BC1,BC2,BC3,CS1a,BC4,CS1d,CS1b,CS1c
BC1,BC2,BC3,CS1a,BC4,CS1d,CS1c,CS1b
BC1,BC2,BC3,CS1b,CS1a,CS1c,BC4,CS1d
BC1,BC2,BC3,CS1b,CS1a,BC4,CS1c,CS1d
BC1,BC2,BC3,CS1b,CS1a,BC4,CS1d,CS1c
BC1,BC2,BC3,CS1b,CS1c,CS1a,BC4,CS1d
BC1,BC2,BC3,CS1b,CS1c,BC4,CS1a,CS1d
BC1,BC2,BC3,CS1b,CS1c,BC4,CS1d,CS1a
BC1,BC2,BC3,CS1b,BC4,CS1a,CS1c,CS1d
BC1,BC2,BC3,CS1b,BC4,CS1a,CS1d,CS1c
BC1,BC2,BC3,CS1b,BC4,CS1c,CS1a,CS1d
BC1,BC2,BC3,CS1b,BC4,CS1c,CS1d,CS1a
BC1,BC2,BC3,CS1b,BC4,CS1d,CS1a,CS1c
BC1,BC2,BC3,CS1b,BC4,CS1d,CS1c,CS1a
BC1,BC2,BC3,CS1c,CS1a,CS1b,BC4,CS1d
BC1,BC2,BC3,CS1c,CS1a,BC4,CS1b,CS1d
BC1,BC2,BC3,CS1c,CS1a,BC4,CS1d,CS1b
BC1,BC2,BC3,CS1c,CS1b,CS1a,BC4,CS1d
BC1,BC2,BC3,CS1c,CS1b,BC4,CS1a,CS1d
BC1,BC2,BC3,CS1c,CS1b,BC4,CS1d,CS1a

BC1,BC2,BC3,CS1c,BC4,CS1a,CS1b,CS1d
BC1,BC2,BC3,CS1c,BC4,CS1a,CS1d,CS1b
BC1,BC2,BC3,CS1c,BC4,CS1b,CS1a,CS1d
BC1,BC2,BC3,CS1c,BC4,CS1b,CS1d,CS1a
BC1,BC2,BC3,CS1c,BC4,CS1d,CS1a,CS1b
BC1,BC2,BC3,CS1c,BC4,CS1d,CS1b,CS1a
BC1,BC2,BC3,BC4,CS1a,CS1b,CS1c,CS1d
BC1,BC2,BC3,BC4,CS1a,CS1b,CS1d,CS1c
BC1,BC2,BC3,BC4,CS1a,CS1c,CS1b,CS1d
BC1,BC2,BC3,BC4,CS1a,CS1c,CS1d,CS1b
BC1,BC2,BC3,BC4,CS1a,CS1d,CS1b,CS1c
BC1,BC2,BC3,BC4,CS1a,CS1d,CS1c,CS1b
BC1,BC2,BC3,BC4,CS1b,CS1a,CS1c,CS1d
BC1,BC2,BC3,BC4,CS1b,CS1a,CS1d,CS1c
BC1,BC2,BC3,BC4,CS1b,CS1c,CS1a,CS1d
BC1,BC2,BC3,BC4,CS1b,CS1c,CS1d,CS1a
BC1,BC2,BC3,BC4,CS1b,CS1d,CS1a,CS1c
BC1,BC2,BC3,BC4,CS1b,CS1d,CS1c,CS1a
BC1,BC2,BC3,BC4,CS1c,CS1a,CS1b,CS1d
BC1,BC2,BC3,BC4,CS1c,CS1a,CS1d,CS1b
BC1,BC2,BC3,BC4,CS1c,CS1b,CS1a,CS1d
BC1,BC2,BC3,BC4,CS1c,CS1b,CS1d,CS1a
BC1,BC2,BC3,BC4,CS1c,CS1d,CS1a,CS1b
BC1,BC2,BC3,BC4,CS1c,CS1d,CS1b,CS1a
BC1,BC2,BC3,BC4,CS1d,CS1a,CS1b,CS1c
BC1,BC2,BC3,BC4,CS1d,CS1a,CS1c,CS1b
BC1,BC2,BC3,BC4,CS1d,CS1b,CS1a,CS1c
BC1,BC2,BC3,BC4,CS1d,CS1b,CS1c,CS1a
BC1,BC2,BC3,BC4,CS1d,CS1c,CS1a,CS1b
BC1,BC2,BC3,BC4,CS1d,CS1c,CS1b,CS1a
BC1,BC2,BC4,CS1a,CS1b,BC3,CS1c,CS1d
BC1,BC2,BC4,CS1a,CS1b,BC3,CS1d,CS1c
BC1,BC2,BC4,CS1a,CS1b,CS1d,BC3,CS1c
BC1,BC2,BC4,CS1a,BC3,CS1b,CS1c,CS1d
BC1,BC2,BC4,CS1a,BC3,CS1b,CS1d,CS1c
BC1,BC2,BC4,CS1a,BC3,CS1c,CS1b,CS1d
BC1,BC2,BC4,CS1a,BC3,CS1c,CS1d,CS1b
BC1,BC2,BC4,CS1a,BC3,CS1d,CS1b,CS1c
BC1,BC2,BC4,CS1a,BC3,CS1d,CS1c,CS1b
BC1,BC2,BC4,CS1a,CS1d,CS1b,BC3,CS1c
BC1,BC2,BC4,CS1a,CS1d,BC3,CS1b,CS1c
BC1,BC2,BC4,CS1a,CS1d,BC3,CS1c,CS1b
BC1,BC2,BC4,CS1b,CS1a,BC3,CS1c,CS1d
BC1,BC2,BC4,CS1b,CS1a,BC3,CS1d,CS1c
BC1,BC2,BC4,CS1b,CS1a,CS1d,BC3,CS1c
BC1,BC2,BC4,CS1b,BC3,CS1a,CS1c,CS1d

BC1,BC2,BC4,CS1b,BC3,CS1a,CS1d,CS1c
BC1,BC2,BC4,CS1b,BC3,CS1c,CS1a,CS1d
BC1,BC2,BC4,CS1b,BC3,CS1c,CS1d,CS1a
BC1,BC2,BC4,CS1b,BC3,CS1d,CS1a,CS1c
BC1,BC2,BC4,CS1b,BC3,CS1d,CS1c,CS1a
BC1,BC2,BC4,CS1b,CS1d,CS1a,BC3,CS1c
BC1,BC2,BC4,CS1b,CS1d,BC3,CS1a,CS1c
BC1,BC2,BC4,CS1b,CS1d,BC3,CS1c,CS1a
BC1,BC2,BC4,BC3,CS1a,CS1b,CS1c,CS1d
BC1,BC2,BC4,BC3,CS1a,CS1b,CS1d,CS1c
BC1,BC2,BC4,BC3,CS1a,CS1c,CS1b,CS1d
BC1,BC2,BC4,BC3,CS1a,CS1c,CS1d,CS1b
BC1,BC2,BC4,BC3,CS1a,CS1d,CS1b,CS1c
BC1,BC2,BC4,BC3,CS1a,CS1d,CS1c,CS1b
BC1,BC2,BC4,BC3,CS1b,CS1a,CS1c,CS1d
BC1,BC2,BC4,BC3,CS1b,CS1a,CS1d,CS1c
BC1,BC2,BC4,BC3,CS1b,CS1c,CS1a,CS1d
BC1,BC2,BC4,BC3,CS1b,CS1c,CS1d,CS1a
BC1,BC2,BC4,BC3,CS1b,CS1d,CS1a,CS1c
BC1,BC2,BC4,BC3,CS1b,CS1d,CS1c,CS1a
BC1,BC2,BC4,BC3,CS1c,CS1a,CS1b,CS1d
BC1,BC2,BC4,BC3,CS1c,CS1a,CS1d,CS1b
BC1,BC2,BC4,BC3,CS1c,CS1b,CS1a,CS1d
BC1,BC2,BC4,BC3,CS1c,CS1b,CS1d,CS1a
BC1,BC2,BC4,BC3,CS1c,CS1d,CS1a,CS1b
BC1,BC2,BC4,BC3,CS1c,CS1d,CS1b,CS1a
BC1,BC2,BC4,BC3,CS1d,CS1a,CS1b,CS1c
BC1,BC2,BC4,BC3,CS1d,CS1a,CS1c,CS1b
BC1,BC2,BC4,BC3,CS1d,CS1b,CS1a,CS1c
BC1,BC2,BC4,BC3,CS1d,CS1b,CS1c,CS1a
BC1,BC2,BC4,BC3,CS1d,CS1c,CS1a,CS1b
BC1,BC2,BC4,BC3,CS1d,CS1c,CS1b,CS1a
BC1,BC2,BC4,CS1d,CS1a,CS1b,BC3,CS1c
BC1,BC2,BC4,CS1d,CS1a,BC3,CS1b,CS1c
BC1,BC2,BC4,CS1d,CS1a,BC3,CS1c,CS1b
BC1,BC2,BC4,CS1d,CS1b,CS1a,BC3,CS1c
BC1,BC2,BC4,CS1d,CS1b,BC3,CS1a,CS1c
BC1,BC2,BC4,CS1d,CS1b,BC3,CS1c,CS1a
BC1,BC2,BC4,CS1d,BC3,CS1a,CS1b,CS1c
BC1,BC2,BC4,CS1d,BC3,CS1a,CS1c,CS1b
BC1,BC2,BC4,CS1d,BC3,CS1b,CS1a,CS1c
BC1,BC2,BC4,CS1d,BC3,CS1b,CS1c,CS1a
BC1,BC2,BC4,CS1d,BC3,CS1c,CS1a,CS1b
BC1,BC2,BC4,CS1d,BC3,CS1c,CS1b,CS1a
BC1,BC3,CS1a,BC2,CS1b,CS1c,BC4,CS1d
BC1,BC3,CS1a,BC2,CS1b,BC4,CS1c,CS1d

BC1,BC3,CS1a,BC2,CS1b,BC4,CS1d,CS1c
BC1,BC3,CS1a,BC2,CS1c,CS1b,BC4,CS1d
BC1,BC3,CS1a,BC2,CS1c,BC4,CS1b,CS1d
BC1,BC3,CS1a,BC2,CS1c,BC4,CS1d,CS1b
BC1,BC3,CS1a,BC2,BC4,CS1b,CS1c,CS1d
BC1,BC3,CS1a,BC2,BC4,CS1b,CS1d,CS1c
BC1,BC3,CS1a,BC2,BC4,CS1c,CS1b,CS1d
BC1,BC3,CS1a,BC2,BC4,CS1c,CS1d,CS1b
BC1,BC3,CS1a,BC2,BC4,CS1d,CS1b,CS1c
BC1,BC3,CS1a,BC2,BC4,CS1d,CS1c,CS1b
BC1,BC3,CS1a,CS1c,BC2,CS1b,BC4,CS1d
BC1,BC3,CS1a,CS1c,BC2,BC4,CS1b,CS1d
BC1,BC3,CS1a,CS1c,BC2,BC4,CS1d,CS1b
BC1,BC3,CS1a,CS1c,BC4,BC2,CS1b,CS1d
BC1,BC3,CS1a,CS1c,BC4,BC2,CS1d,CS1b
BC1,BC3,CS1a,CS1c,BC4,CS1d,BC2,CS1b
BC1,BC3,CS1a,BC4,BC2,CS1b,CS1c,CS1d
BC1,BC3,CS1a,BC4,BC2,CS1b,CS1d,CS1c
BC1,BC3,CS1a,BC4,BC2,CS1c,CS1b,CS1d
BC1,BC3,CS1a,BC4,BC2,CS1c,CS1d,CS1b
BC1,BC3,CS1a,BC4,BC2,CS1d,CS1b,CS1c
BC1,BC3,CS1a,BC4,BC2,CS1d,CS1c,CS1b
BC1,BC3,CS1a,BC4,CS1c,BC2,CS1b,CS1d
BC1,BC3,CS1a,BC4,CS1c,BC2,CS1d,CS1b
BC1,BC3,CS1a,BC4,CS1c,CS1d,BC2,CS1b
BC1,BC3,CS1a,BC4,CS1d,BC2,CS1b,CS1c
BC1,BC3,CS1a,BC4,CS1d,BC2,CS1c,CS1b
BC1,BC3,CS1a,BC4,CS1d,CS1c,BC2,CS1b
BC1,BC3,BC2,CS1a,CS1b,CS1c,BC4,CS1d
BC1,BC3,BC2,CS1a,CS1b,BC4,CS1c,CS1d
BC1,BC3,BC2,CS1a,CS1b,BC4,CS1d,CS1c
BC1,BC3,BC2,CS1a,CS1c,CS1b,BC4,CS1d
BC1,BC3,BC2,CS1a,CS1c,BC4,CS1b,CS1d
BC1,BC3,BC2,CS1a,CS1c,BC4,CS1d,CS1b
BC1,BC3,BC2,CS1a,BC4,CS1b,CS1c,CS1d
BC1,BC3,BC2,CS1a,BC4,CS1b,CS1d,CS1c
BC1,BC3,BC2,CS1a,BC4,CS1c,CS1b,CS1d
BC1,BC3,BC2,CS1a,BC4,CS1c,CS1d,CS1b
BC1,BC3,BC2,CS1a,BC4,CS1d,CS1b,CS1c
BC1,BC3,BC2,CS1a,BC4,CS1d,CS1c,CS1b
BC1,BC3,BC2,CS1b,CS1a,CS1c,BC4,CS1d
BC1,BC3,BC2,CS1b,CS1a,BC4,CS1c,CS1d
BC1,BC3,BC2,CS1b,CS1a,BC4,CS1d,CS1c
BC1,BC3,BC2,CS1b,CS1c,CS1a,BC4,CS1d
BC1,BC3,BC2,CS1b,CS1c,BC4,CS1a,CS1d
BC1,BC3,BC2,CS1b,CS1c,BC4,CS1d,CS1a

BC1,BC3,BC2,CS1b,BC4,CS1a,CS1c,CS1d
BC1,BC3,BC2,CS1b,BC4,CS1a,CS1d,CS1c
BC1,BC3,BC2,CS1b,BC4,CS1c,CS1a,CS1d
BC1,BC3,BC2,CS1b,BC4,CS1c,CS1d,CS1a
BC1,BC3,BC2,CS1b,BC4,CS1d,CS1a,CS1c
BC1,BC3,BC2,CS1b,BC4,CS1d,CS1c,CS1a
BC1,BC3,BC2,CS1c,CS1a,CS1b,BC4,CS1d
BC1,BC3,BC2,CS1c,CS1a,BC4,CS1b,CS1d
BC1,BC3,BC2,CS1c,CS1a,BC4,CS1d,CS1b
BC1,BC3,BC2,CS1c,CS1b,CS1a,BC4,CS1d
BC1,BC3,BC2,CS1c,CS1b,BC4,CS1a,CS1d
BC1,BC3,BC2,CS1c,CS1b,BC4,CS1d,CS1a
BC1,BC3,BC2,CS1c,BC4,CS1a,CS1b,CS1d
BC1,BC3,BC2,CS1c,BC4,CS1a,CS1d,CS1b
BC1,BC3,BC2,CS1c,BC4,CS1b,CS1a,CS1d
BC1,BC3,BC2,CS1c,BC4,CS1b,CS1d,CS1a
BC1,BC3,BC2,CS1c,BC4,CS1d,CS1a,CS1b
BC1,BC3,BC2,CS1c,BC4,CS1d,CS1b,CS1a
BC1,BC3,BC2,BC4,CS1a,CS1b,CS1c,CS1d
BC1,BC3,BC2,BC4,CS1a,CS1b,CS1d,CS1c
BC1,BC3,BC2,BC4,CS1a,CS1c,CS1b,CS1d
BC1,BC3,BC2,BC4,CS1a,CS1c,CS1d,CS1b
BC1,BC3,BC2,BC4,CS1a,CS1d,CS1b,CS1c
BC1,BC3,BC2,BC4,CS1a,CS1d,CS1c,CS1b
BC1,BC3,BC2,BC4,CS1b,CS1a,CS1c,CS1d
BC1,BC3,BC2,BC4,CS1b,CS1a,CS1d,CS1c
BC1,BC3,BC2,BC4,CS1b,CS1c,CS1a,CS1d
BC1,BC3,BC2,BC4,CS1b,CS1c,CS1d,CS1a
BC1,BC3,BC2,BC4,CS1b,CS1d,CS1a,CS1c
BC1,BC3,BC2,BC4,CS1b,CS1d,CS1c,CS1a
BC1,BC3,BC2,BC4,CS1c,CS1a,CS1b,CS1d
BC1,BC3,BC2,BC4,CS1c,CS1a,CS1d,CS1b
BC1,BC3,BC2,BC4,CS1c,CS1b,CS1a,CS1d
BC1,BC3,BC2,BC4,CS1c,CS1b,CS1d,CS1a
BC1,BC3,BC2,BC4,CS1c,CS1d,CS1a,CS1b
BC1,BC3,BC2,BC4,CS1c,CS1d,CS1b,CS1a
BC1,BC3,BC2,BC4,CS1d,CS1a,CS1b,CS1c
BC1,BC3,BC2,BC4,CS1d,CS1a,CS1c,CS1b
BC1,BC3,BC2,BC4,CS1d,CS1b,CS1a,CS1c
BC1,BC3,BC2,BC4,CS1d,CS1b,CS1c,CS1a
BC1,BC3,BC2,BC4,CS1d,CS1c,CS1a,CS1b
BC1,BC3,BC2,BC4,CS1d,CS1c,CS1b,CS1a
BC1,BC3,CS1c,CS1a,BC2,CS1b,BC4,CS1d
BC1,BC3,CS1c,CS1a,BC2,BC4,CS1b,CS1d
BC1,BC3,CS1c,CS1a,BC2,BC4,CS1d,CS1b
BC1,BC3,CS1c,CS1a,BC4,BC2,CS1b,CS1d

BC1,BC3,CS1c,CS1a,BC4,BC2,CS1d,CS1b
BC1,BC3,CS1c,CS1a,BC4,CS1d,BC2,CS1b
BC1,BC3,CS1c,BC2,CS1a,CS1b,BC4,CS1d
BC1,BC3,CS1c,BC2,CS1a,BC4,CS1b,CS1d
BC1,BC3,CS1c,BC2,CS1a,BC4,CS1d,CS1b
BC1,BC3,CS1c,BC2,CS1b,CS1a,BC4,CS1d
BC1,BC3,CS1c,BC2,CS1b,BC4,CS1a,CS1d
BC1,BC3,CS1c,BC2,CS1b,BC4,CS1d,CS1a
BC1,BC3,CS1c,BC2,BC4,CS1a,CS1b,CS1d
BC1,BC3,CS1c,BC2,BC4,CS1a,CS1d,CS1b
BC1,BC3,CS1c,BC2,BC4,CS1b,CS1a,CS1d
BC1,BC3,CS1c,BC2,BC4,CS1b,CS1d,CS1a
BC1,BC3,CS1c,BC2,BC4,CS1d,CS1a,CS1b
BC1,BC3,CS1c,BC2,BC4,CS1d,CS1b,CS1a
BC1,BC3,CS1c,BC4,CS1a,BC2,CS1b,CS1d
BC1,BC3,CS1c,BC4,CS1a,BC2,CS1d,CS1b
BC1,BC3,CS1c,BC4,CS1a,CS1d,BC2,CS1b
BC1,BC3,CS1c,BC4,BC2,CS1a,CS1b,CS1d
BC1,BC3,CS1c,BC4,BC2,CS1a,CS1d,CS1b
BC1,BC3,CS1c,BC4,BC2,CS1b,CS1a,CS1d
BC1,BC3,CS1c,BC4,BC2,CS1b,CS1d,CS1a
BC1,BC3,CS1c,BC4,BC2,CS1d,CS1a,CS1b
BC1,BC3,CS1c,BC4,BC2,CS1d,CS1b,CS1a
BC1,BC3,CS1c,BC4,CS1d,CS1a,BC2,CS1b
BC1,BC3,CS1c,BC4,CS1d,BC2,CS1a,CS1b
BC1,BC3,CS1c,BC4,CS1d,BC2,CS1b,CS1a
BC1,BC3,BC4,CS1a,BC2,CS1b,CS1c,CS1d
BC1,BC3,BC4,CS1a,BC2,CS1b,CS1d,CS1c
BC1,BC3,BC4,CS1a,BC2,CS1c,CS1b,CS1d
BC1,BC3,BC4,CS1a,BC2,CS1c,CS1d,CS1b
BC1,BC3,BC4,CS1a,BC2,CS1d,CS1b,CS1c
BC1,BC3,BC4,CS1a,BC2,CS1d,CS1c,CS1b
BC1,BC3,BC4,CS1a,CS1c,BC2,CS1b,CS1d
BC1,BC3,BC4,CS1a,CS1c,BC2,CS1d,CS1b
BC1,BC3,BC4,CS1a,CS1c,CS1d,BC2,CS1b
BC1,BC3,BC4,CS1a,CS1d,BC2,CS1b,CS1c
BC1,BC3,BC4,CS1a,CS1d,BC2,CS1c,CS1b
BC1,BC3,BC4,CS1a,CS1d,CS1c,BC2,CS1b
BC1,BC3,BC4,BC2,CS1a,CS1b,CS1c,CS1d
BC1,BC3,BC4,BC2,CS1a,CS1b,CS1d,CS1c
BC1,BC3,BC4,BC2,CS1a,CS1c,CS1b,CS1d
BC1,BC3,BC4,BC2,CS1a,CS1c,CS1d,CS1b
BC1,BC3,BC4,BC2,CS1a,CS1d,CS1b,CS1c
BC1,BC3,BC4,BC2,CS1a,CS1d,CS1c,CS1b
BC1,BC3,BC4,BC2,CS1b,CS1a,CS1c,CS1d
BC1,BC3,BC4,BC2,CS1b,CS1a,CS1d,CS1c

BC1,BC3,BC4,BC2,CS1b,CS1c,CS1a,CS1d
BC1,BC3,BC4,BC2,CS1b,CS1c,CS1d,CS1a
BC1,BC3,BC4,BC2,CS1b,CS1d,CS1a,CS1c
BC1,BC3,BC4,BC2,CS1b,CS1d,CS1c,CS1a
BC1,BC3,BC4,BC2,CS1c,CS1a,CS1b,CS1d
BC1,BC3,BC4,BC2,CS1c,CS1a,CS1d,CS1b
BC1,BC3,BC4,BC2,CS1c,CS1b,CS1a,CS1d
BC1,BC3,BC4,BC2,CS1c,CS1b,CS1d,CS1a
BC1,BC3,BC4,BC2,CS1c,CS1d,CS1a,CS1b
BC1,BC3,BC4,BC2,CS1c,CS1d,CS1b,CS1a
BC1,BC3,BC4,BC2,CS1d,CS1a,CS1b,CS1c
BC1,BC3,BC4,BC2,CS1d,CS1a,CS1c,CS1b
BC1,BC3,BC4,BC2,CS1d,CS1b,CS1a,CS1c
BC1,BC3,BC4,BC2,CS1d,CS1b,CS1c,CS1a
BC1,BC3,BC4,BC2,CS1d,CS1c,CS1a,CS1b
BC1,BC3,BC4,BC2,CS1d,CS1c,CS1b,CS1a
BC1,BC3,BC4,CS1c,CS1a,BC2,CS1b,CS1d
BC1,BC3,BC4,CS1c,CS1a,BC2,CS1d,CS1b
BC1,BC3,BC4,CS1c,CS1a,CS1d,BC2,CS1b
BC1,BC3,BC4,CS1c,BC2,CS1a,CS1b,CS1d
BC1,BC3,BC4,CS1c,BC2,CS1a,CS1d,CS1b
BC1,BC3,BC4,CS1c,BC2,CS1b,CS1a,CS1d
BC1,BC3,BC4,CS1c,BC2,CS1b,CS1d,CS1a
BC1,BC3,BC4,CS1c,BC2,CS1d,CS1a,CS1b
BC1,BC3,BC4,CS1c,BC2,CS1d,CS1b,CS1a
BC1,BC3,BC4,CS1c,CS1d,CS1a,BC2,CS1b
BC1,BC3,BC4,CS1c,CS1d,BC2,CS1a,CS1b
BC1,BC3,BC4,CS1c,CS1d,BC2,CS1b,CS1a
BC1,BC3,BC4,CS1d,CS1a,BC2,CS1b,CS1c
BC1,BC3,BC4,CS1d,CS1a,BC2,CS1c,CS1b
BC1,BC3,BC4,CS1d,CS1a,CS1c,BC2,CS1b
BC1,BC3,BC4,CS1d,BC2,CS1a,CS1b,CS1c
BC1,BC3,BC4,CS1d,BC2,CS1a,CS1c,CS1b
BC1,BC3,BC4,CS1d,BC2,CS1b,CS1a,CS1c
BC1,BC3,BC4,CS1d,BC2,CS1b,CS1c,CS1a
BC1,BC3,BC4,CS1d,BC2,CS1c,CS1a,CS1b
BC1,BC3,BC4,CS1d,BC2,CS1c,CS1b,CS1a
BC1,BC3,BC4,CS1d,CS1c,CS1a,BC2,CS1b
BC1,BC3,BC4,CS1d,CS1c,BC2,CS1a,CS1b
BC1,BC3,BC4,CS1d,CS1c,BC2,CS1b,CS1a
BC1,BC4,CS1a,BC2,CS1b,BC3,CS1c,CS1d
BC1,BC4,CS1a,BC2,CS1b,BC3,CS1d,CS1c
BC1,BC4,CS1a,BC2,CS1b,CS1d,BC3,CS1c
BC1,BC4,CS1a,BC2,BC3,CS1b,CS1c,CS1d
BC1,BC4,CS1a,BC2,BC3,CS1b,CS1d,CS1c
BC1,BC4,CS1a,BC2,BC3,CS1c,CS1b,CS1d

BC1,BC4,CS1a,BC2,BC3,CS1c,CS1d,CS1b
BC1,BC4,CS1a,BC2,BC3,CS1d,CS1b,CS1c
BC1,BC4,CS1a,BC2,BC3,CS1d,CS1c,CS1b
BC1,BC4,CS1a,BC2,CS1d,CS1b,BC3,CS1c
BC1,BC4,CS1a,BC2,CS1d,BC3,CS1b,CS1c
BC1,BC4,CS1a,BC2,CS1d,BC3,CS1c,CS1b
BC1,BC4,CS1a,BC3,BC2,CS1b,CS1c,CS1d
BC1,BC4,CS1a,BC3,BC2,CS1b,CS1d,CS1c
BC1,BC4,CS1a,BC3,BC2,CS1c,CS1b,CS1d
BC1,BC4,CS1a,BC3,BC2,CS1c,CS1d,CS1b
BC1,BC4,CS1a,BC3,BC2,CS1d,CS1b,CS1c
BC1,BC4,CS1a,BC3,BC2,CS1d,CS1c,CS1b
BC1,BC4,CS1a,BC3,CS1c,BC2,CS1b,CS1d
BC1,BC4,CS1a,BC3,CS1c,BC2,CS1d,CS1b
BC1,BC4,CS1a,BC3,CS1c,CS1d,BC2,CS1b
BC1,BC4,CS1a,BC3,CS1d,BC2,CS1b,CS1c
BC1,BC4,CS1a,BC3,CS1d,BC2,CS1c,CS1b
BC1,BC4,CS1a,BC3,CS1d,CS1c,BC2,CS1b
BC1,BC4,CS1a,CS1d,BC2,CS1b,BC3,CS1c
BC1,BC4,CS1a,CS1d,BC2,BC3,CS1b,CS1c
BC1,BC4,CS1a,CS1d,BC2,BC3,CS1c,CS1b
BC1,BC4,CS1a,CS1d,BC3,BC2,CS1b,CS1c
BC1,BC4,CS1a,CS1d,BC3,BC2,CS1c,CS1b
BC1,BC4,CS1a,CS1d,BC3,CS1c,BC2,CS1b
BC1,BC4,BC2,CS1a,CS1b,BC3,CS1c,CS1d
BC1,BC4,BC2,CS1a,CS1b,BC3,CS1d,CS1c
BC1,BC4,BC2,CS1a,CS1b,CS1d,BC3,CS1c
BC1,BC4,BC2,CS1a,BC3,CS1b,CS1c,CS1d
BC1,BC4,BC2,CS1a,BC3,CS1b,CS1d,CS1c
BC1,BC4,BC2,CS1a,BC3,CS1c,CS1b,CS1d
BC1,BC4,BC2,CS1a,BC3,CS1c,CS1d,CS1b
BC1,BC4,BC2,CS1a,BC3,CS1d,CS1b,CS1c
BC1,BC4,BC2,CS1a,BC3,CS1d,CS1c,CS1b
BC1,BC4,BC2,CS1a,CS1d,CS1b,BC3,CS1c
BC1,BC4,BC2,CS1a,CS1d,BC3,CS1b,CS1c
BC1,BC4,BC2,CS1a,CS1d,BC3,CS1c,CS1b
BC1,BC4,BC2,CS1b,CS1a,BC3,CS1c,CS1d
BC1,BC4,BC2,CS1b,CS1a,BC3,CS1d,CS1c
BC1,BC4,BC2,CS1b,CS1a,CS1d,BC3,CS1c
BC1,BC4,BC2,CS1b,BC3,CS1a,CS1c,CS1d
BC1,BC4,BC2,CS1b,BC3,CS1a,CS1d,CS1c
BC1,BC4,BC2,CS1b,BC3,CS1c,CS1a,CS1d
BC1,BC4,BC2,CS1b,BC3,CS1c,CS1d,CS1a
BC1,BC4,BC2,CS1b,BC3,CS1d,CS1a,CS1c
BC1,BC4,BC2,CS1b,BC3,CS1d,CS1c,CS1a
BC1,BC4,BC2,CS1b,CS1d,CS1a,BC3,CS1c

BC1,BC4,BC2,CS1b,CS1d,BC3,CS1a,CS1c
BC1,BC4,BC2,CS1b,CS1d,BC3,CS1c,CS1a
BC1,BC4,BC2,BC3,CS1a,CS1b,CS1c,CS1d
BC1,BC4,BC2,BC3,CS1a,CS1b,CS1d,CS1c
BC1,BC4,BC2,BC3,CS1a,CS1c,CS1b,CS1d
BC1,BC4,BC2,BC3,CS1a,CS1c,CS1d,CS1b
BC1,BC4,BC2,BC3,CS1a,CS1d,CS1b,CS1c
BC1,BC4,BC2,BC3,CS1a,CS1d,CS1c,CS1b
BC1,BC4,BC2,BC3,CS1b,CS1a,CS1c,CS1d
BC1,BC4,BC2,BC3,CS1b,CS1a,CS1d,CS1c
BC1,BC4,BC2,BC3,CS1b,CS1c,CS1a,CS1d
BC1,BC4,BC2,BC3,CS1b,CS1c,CS1d,CS1a
BC1,BC4,BC2,BC3,CS1b,CS1d,CS1a,CS1c
BC1,BC4,BC2,BC3,CS1b,CS1d,CS1c,CS1a
BC1,BC4,BC2,BC3,CS1c,CS1a,CS1b,CS1d
BC1,BC4,BC2,BC3,CS1c,CS1a,CS1d,CS1b
BC1,BC4,BC2,BC3,CS1c,CS1b,CS1a,CS1d
BC1,BC4,BC2,BC3,CS1c,CS1b,CS1d,CS1a
BC1,BC4,BC2,BC3,CS1c,CS1d,CS1a,CS1b
BC1,BC4,BC2,BC3,CS1c,CS1d,CS1b,CS1a
BC1,BC4,BC2,BC3,CS1d,CS1a,CS1b,CS1c
BC1,BC4,BC2,BC3,CS1d,CS1a,CS1c,CS1b
BC1,BC4,BC2,BC3,CS1d,CS1b,CS1a,CS1c
BC1,BC4,BC2,BC3,CS1d,CS1b,CS1c,CS1a
BC1,BC4,BC2,BC3,CS1d,CS1c,CS1a,CS1b
BC1,BC4,BC2,BC3,CS1d,CS1c,CS1b,CS1a
BC1,BC4,BC2,CS1d,CS1a,CS1b,BC3,CS1c
BC1,BC4,BC2,CS1d,CS1a,BC3,CS1b,CS1c
BC1,BC4,BC2,CS1d,CS1a,BC3,CS1c,CS1b
BC1,BC4,BC2,CS1d,CS1b,CS1a,BC3,CS1c
BC1,BC4,BC2,CS1d,CS1b,BC3,CS1a,CS1c
BC1,BC4,BC2,CS1d,CS1b,BC3,CS1c,CS1a
BC1,BC4,BC2,CS1d,BC3,CS1a,CS1b,CS1c
BC1,BC4,BC2,CS1d,BC3,CS1a,CS1c,CS1b
BC1,BC4,BC2,CS1d,BC3,CS1b,CS1a,CS1c
BC1,BC4,BC2,CS1d,BC3,CS1b,CS1c,CS1a
BC1,BC4,BC2,CS1d,BC3,CS1c,CS1a,CS1b
BC1,BC4,BC2,CS1d,BC3,CS1c,CS1b,CS1a
BC1,BC4,BC3,CS1a,BC2,CS1b,CS1c,CS1d
BC1,BC4,BC3,CS1a,BC2,CS1b,CS1d,CS1c
BC1,BC4,BC3,CS1a,BC2,CS1c,CS1b,CS1d
BC1,BC4,BC3,CS1a,BC2,CS1c,CS1d,CS1b
BC1,BC4,BC3,CS1a,BC2,CS1d,CS1b,CS1c
BC1,BC4,BC3,CS1a,BC2,CS1d,CS1c,CS1b
BC1,BC4,BC3,CS1a,CS1c,BC2,CS1b,CS1d
BC1,BC4,BC3,CS1a,CS1c,BC2,CS1d,CS1b

BC1,BC4,BC3,CS1a,CS1c,CS1d,BC2,CS1b
BC1,BC4,BC3,CS1a,CS1d,BC2,CS1b,CS1c
BC1,BC4,BC3,CS1a,CS1d,BC2,CS1c,CS1b
BC1,BC4,BC3,CS1a,CS1d,CS1c,BC2,CS1b
BC1,BC4,BC3,BC2,CS1a,CS1b,CS1c,CS1d
BC1,BC4,BC3,BC2,CS1a,CS1b,CS1d,CS1c
BC1,BC4,BC3,BC2,CS1a,CS1c,CS1b,CS1d
BC1,BC4,BC3,BC2,CS1a,CS1c,CS1d,CS1b
BC1,BC4,BC3,BC2,CS1a,CS1d,CS1b,CS1c
BC1,BC4,BC3,BC2,CS1a,CS1d,CS1c,CS1b
BC1,BC4,BC3,BC2,CS1b,CS1a,CS1c,CS1d
BC1,BC4,BC3,BC2,CS1b,CS1a,CS1d,CS1c
BC1,BC4,BC3,BC2,CS1b,CS1c,CS1a,CS1d
BC1,BC4,BC3,BC2,CS1b,CS1c,CS1d,CS1a
BC1,BC4,BC3,BC2,CS1b,CS1d,CS1a,CS1c
BC1,BC4,BC3,BC2,CS1b,CS1d,CS1c,CS1a
BC1,BC4,BC3,BC2,CS1c,CS1a,CS1b,CS1d
BC1,BC4,BC3,BC2,CS1c,CS1a,CS1d,CS1b
BC1,BC4,BC3,BC2,CS1c,CS1b,CS1a,CS1d
BC1,BC4,BC3,BC2,CS1c,CS1b,CS1d,CS1a
BC1,BC4,BC3,BC2,CS1c,CS1d,CS1a,CS1b
BC1,BC4,BC3,BC2,CS1c,CS1d,CS1b,CS1a
BC1,BC4,BC3,BC2,CS1d,CS1a,CS1b,CS1c
BC1,BC4,BC3,BC2,CS1d,CS1a,CS1c,CS1b
BC1,BC4,BC3,BC2,CS1d,CS1b,CS1a,CS1c
BC1,BC4,BC3,BC2,CS1d,CS1b,CS1c,CS1a
BC1,BC4,BC3,BC2,CS1d,CS1c,CS1a,CS1b
BC1,BC4,BC3,BC2,CS1d,CS1c,CS1b,CS1a
BC1,BC4,BC3,CS1c,CS1a,BC2,CS1b,CS1d
BC1,BC4,BC3,CS1c,CS1a,BC2,CS1d,CS1b
BC1,BC4,BC3,CS1c,CS1a,CS1d,BC2,CS1b
BC1,BC4,BC3,CS1c,BC2,CS1a,CS1b,CS1d
BC1,BC4,BC3,CS1c,BC2,CS1a,CS1d,CS1b
BC1,BC4,BC3,CS1c,BC2,CS1b,CS1a,CS1d
BC1,BC4,BC3,CS1c,BC2,CS1b,CS1d,CS1a
BC1,BC4,BC3,CS1c,BC2,CS1d,CS1a,CS1b
BC1,BC4,BC3,CS1c,BC2,CS1d,CS1b,CS1a
BC1,BC4,BC3,CS1c,CS1d,CS1a,BC2,CS1b
BC1,BC4,BC3,CS1c,CS1d,BC2,CS1a,CS1b
BC1,BC4,BC3,CS1c,CS1d,BC2,CS1b,CS1a
BC1,BC4,BC3,CS1d,CS1a,BC2,CS1b,CS1c
BC1,BC4,BC3,CS1d,CS1a,BC2,CS1c,CS1b
BC1,BC4,BC3,CS1d,CS1a,CS1c,BC2,CS1b
BC1,BC4,BC3,CS1d,BC2,CS1a,CS1b,CS1c
BC1,BC4,BC3,CS1d,BC2,CS1a,CS1c,CS1b
BC1,BC4,BC3,CS1d,BC2,CS1b,CS1a,CS1c

BC1,BC4,BC3,CS1d,BC2,CS1b,CS1c,CS1a
BC1,BC4,BC3,CS1d,BC2,CS1c,CS1a,CS1b
BC1,BC4,BC3,CS1d,BC2,CS1c,CS1b,CS1a
BC1,BC4,BC3,CS1d,CS1c,CS1a,BC2,CS1b
BC1,BC4,BC3,CS1d,CS1c,BC2,CS1a,CS1b
BC1,BC4,BC3,CS1d,CS1c,BC2,CS1b,CS1a
BC1,BC4,CS1d,CS1a,BC2,CS1b,BC3,CS1c
BC1,BC4,CS1d,CS1a,BC2,BC3,CS1b,CS1c
BC1,BC4,CS1d,CS1a,BC2,BC3,CS1c,CS1b
BC1,BC4,CS1d,CS1a,BC3,BC2,CS1b,CS1c
BC1,BC4,CS1d,CS1a,BC3,BC2,CS1c,CS1b
BC1,BC4,CS1d,CS1a,BC3,CS1c,BC2,CS1b
BC1,BC4,CS1d,BC2,CS1a,CS1b,BC3,CS1c
BC1,BC4,CS1d,BC2,CS1a,BC3,CS1b,CS1c
BC1,BC4,CS1d,BC2,CS1a,BC3,CS1c,CS1b
BC1,BC4,CS1d,BC2,CS1b,CS1a,BC3,CS1c
BC1,BC4,CS1d,BC2,CS1b,BC3,CS1a,CS1c
BC1,BC4,CS1d,BC2,CS1b,BC3,CS1c,CS1a
BC1,BC4,CS1d,BC2,BC3,CS1a,CS1b,CS1c
BC1,BC4,CS1d,BC2,BC3,CS1a,CS1c,CS1b
BC1,BC4,CS1d,BC2,BC3,CS1b,CS1a,CS1c
BC1,BC4,CS1d,BC2,BC3,CS1b,CS1c,CS1a
BC1,BC4,CS1d,BC2,BC3,CS1c,CS1a,CS1b
BC1,BC4,CS1d,BC2,BC3,CS1c,CS1b,CS1a
BC1,BC4,CS1d,BC3,CS1a,BC2,CS1b,CS1c
BC1,BC4,CS1d,BC3,CS1a,BC2,CS1c,CS1b
BC1,BC4,CS1d,BC3,CS1a,CS1c,BC2,CS1b
BC1,BC4,CS1d,BC3,BC2,CS1a,CS1b,CS1c
BC1,BC4,CS1d,BC3,BC2,CS1a,CS1c,CS1b
BC1,BC4,CS1d,BC3,BC2,CS1b,CS1a,CS1c
BC1,BC4,CS1d,BC3,BC2,CS1b,CS1c,CS1a
BC1,BC4,CS1d,BC3,BC2,CS1c,CS1a,CS1b
BC1,BC4,CS1d,BC3,BC2,CS1c,CS1b,CS1a
BC1,BC4,CS1d,BC3,CS1c,CS1a,BC2,CS1b
BC1,BC4,CS1d,BC3,CS1c,BC2,CS1a,CS1b
BC1,BC4,CS1d,BC3,CS1c,BC2,CS1b,CS1a
BC2,BC1,CS1a,CS1b,BC3,CS1c,BC4,CS1d
BC2,BC1,CS1a,CS1b,BC3,BC4,CS1c,CS1d
BC2,BC1,CS1a,CS1b,BC3,BC4,CS1d,CS1c
BC2,BC1,CS1a,CS1b,BC4,BC3,CS1c,CS1d
BC2,BC1,CS1a,CS1b,BC4,BC3,CS1d,CS1c
BC2,BC1,CS1a,CS1b,BC4,CS1d,BC3,CS1c
BC2,BC1,CS1a,BC3,CS1b,CS1c,BC4,CS1d
BC2,BC1,CS1a,BC3,CS1b,BC4,CS1c,CS1d
BC2,BC1,CS1a,BC3,CS1b,BC4,CS1d,CS1c
BC2,BC1,CS1a,BC3,CS1c,CS1b,BC4,CS1d

BC2,BC1,CS1a,BC3,CS1c,BC4,CS1b,CS1d
BC2,BC1,CS1a,BC3,CS1c,BC4,CS1d,CS1b
BC2,BC1,CS1a,BC3,BC4,CS1b,CS1c,CS1d
BC2,BC1,CS1a,BC3,BC4,CS1b,CS1d,CS1c
BC2,BC1,CS1a,BC3,BC4,CS1c,CS1b,CS1d
BC2,BC1,CS1a,BC3,BC4,CS1c,CS1d,CS1b
BC2,BC1,CS1a,BC3,BC4,CS1d,CS1b,CS1c
BC2,BC1,CS1a,BC3,BC4,CS1d,CS1c,CS1b
BC2,BC1,CS1a,BC4,CS1b,BC3,CS1c,CS1d
BC2,BC1,CS1a,BC4,CS1b,BC3,CS1d,CS1c
BC2,BC1,CS1a,BC4,CS1b,CS1d,BC3,CS1c
BC2,BC1,CS1a,BC4,BC3,CS1b,CS1c,CS1d
BC2,BC1,CS1a,BC4,BC3,CS1b,CS1d,CS1c
BC2,BC1,CS1a,BC4,BC3,CS1c,CS1b,CS1d
BC2,BC1,CS1a,BC4,BC3,CS1c,CS1d,CS1b
BC2,BC1,CS1a,BC4,BC3,CS1d,CS1b,CS1c
BC2,BC1,CS1a,BC4,BC3,CS1d,CS1c,CS1b
BC2,BC1,CS1a,BC4,CS1d,CS1b,BC3,CS1c
BC2,BC1,CS1a,BC4,CS1d,BC3,CS1b,CS1c
BC2,BC1,CS1a,BC4,CS1d,BC3,CS1c,CS1b
BC2,BC1,CS1b,CS1a,BC3,CS1c,BC4,CS1d
BC2,BC1,CS1b,CS1a,BC3,BC4,CS1c,CS1d
BC2,BC1,CS1b,CS1a,BC3,BC4,CS1d,CS1c
BC2,BC1,CS1b,CS1a,BC4,BC3,CS1c,CS1d
BC2,BC1,CS1b,CS1a,BC4,BC3,CS1d,CS1c
BC2,BC1,CS1b,CS1a,BC4,CS1d,BC3,CS1c
BC2,BC1,CS1b,BC3,CS1a,CS1c,BC4,CS1d
BC2,BC1,CS1b,BC3,CS1a,BC4,CS1c,CS1d
BC2,BC1,CS1b,BC3,CS1a,BC4,CS1d,CS1c
BC2,BC1,CS1b,BC3,CS1c,CS1a,BC4,CS1d
BC2,BC1,CS1b,BC3,CS1c,BC4,CS1a,CS1d
BC2,BC1,CS1b,BC3,CS1c,BC4,CS1d,CS1a
BC2,BC1,CS1b,BC3,BC4,CS1a,CS1c,CS1d
BC2,BC1,CS1b,BC3,BC4,CS1a,CS1d,CS1c
BC2,BC1,CS1b,BC3,BC4,CS1c,CS1a,CS1d
BC2,BC1,CS1b,BC3,BC4,CS1c,CS1d,CS1a
BC2,BC1,CS1b,BC3,BC4,CS1d,CS1a,CS1c
BC2,BC1,CS1b,BC3,BC4,CS1d,CS1c,CS1a
BC2,BC1,CS1b,BC4,CS1a,BC3,CS1c,CS1d
BC2,BC1,CS1b,BC4,CS1a,BC3,CS1d,CS1c
BC2,BC1,CS1b,BC4,CS1a,CS1d,BC3,CS1c
BC2,BC1,CS1b,BC4,BC3,CS1a,CS1c,CS1d
BC2,BC1,CS1b,BC4,BC3,CS1a,CS1d,CS1c
BC2,BC1,CS1b,BC4,BC3,CS1c,CS1a,CS1d
BC2,BC1,CS1b,BC4,BC3,CS1c,CS1d,CS1a
BC2,BC1,CS1b,BC4,BC3,CS1d,CS1a,CS1c

BC2,BC1,CS1b,BC4,BC3,CS1d,CS1c,CS1a
BC2,BC1,CS1b,BC4,CS1d,CS1a,BC3,CS1c
BC2,BC1,CS1b,BC4,CS1d,BC3,CS1a,CS1c
BC2,BC1,CS1b,BC4,CS1d,BC3,CS1c,CS1a
BC2,BC1,BC3,CS1a,CS1b,CS1c,BC4,CS1d
BC2,BC1,BC3,CS1a,CS1b,BC4,CS1c,CS1d
BC2,BC1,BC3,CS1a,CS1b,BC4,CS1d,CS1c
BC2,BC1,BC3,CS1a,CS1c,CS1b,BC4,CS1d
BC2,BC1,BC3,CS1a,CS1c,BC4,CS1b,CS1d
BC2,BC1,BC3,CS1a,CS1c,BC4,CS1d,CS1b
BC2,BC1,BC3,CS1a,BC4,CS1b,CS1c,CS1d
BC2,BC1,BC3,CS1a,BC4,CS1b,CS1d,CS1c
BC2,BC1,BC3,CS1a,BC4,CS1c,CS1b,CS1d
BC2,BC1,BC3,CS1a,BC4,CS1c,CS1d,CS1b
BC2,BC1,BC3,CS1a,BC4,CS1d,CS1b,CS1c
BC2,BC1,BC3,CS1a,BC4,CS1d,CS1c,CS1b
BC2,BC1,BC3,CS1b,CS1a,CS1c,BC4,CS1d
BC2,BC1,BC3,CS1b,CS1a,BC4,CS1c,CS1d
BC2,BC1,BC3,CS1b,CS1a,BC4,CS1d,CS1c
BC2,BC1,BC3,CS1b,CS1c,CS1a,BC4,CS1d
BC2,BC1,BC3,CS1b,CS1c,BC4,CS1a,CS1d
BC2,BC1,BC3,CS1b,CS1c,BC4,CS1d,CS1a
BC2,BC1,BC3,CS1b,BC4,CS1a,CS1c,CS1d
BC2,BC1,BC3,CS1b,BC4,CS1a,CS1d,CS1c
BC2,BC1,BC3,CS1b,BC4,CS1c,CS1a,CS1d
BC2,BC1,BC3,CS1b,BC4,CS1c,CS1d,CS1a
BC2,BC1,BC3,CS1b,BC4,CS1d,CS1a,CS1c
BC2,BC1,BC3,CS1b,BC4,CS1d,CS1c,CS1a
BC2,BC1,BC3,CS1c,CS1a,CS1b,BC4,CS1d
BC2,BC1,BC3,CS1c,CS1a,BC4,CS1b,CS1d
BC2,BC1,BC3,CS1c,CS1a,BC4,CS1d,CS1b
BC2,BC1,BC3,CS1c,CS1b,CS1a,BC4,CS1d
BC2,BC1,BC3,CS1c,CS1b,BC4,CS1a,CS1d
BC2,BC1,BC3,CS1c,CS1b,BC4,CS1d,CS1a
BC2,BC1,BC3,CS1c,BC4,CS1a,CS1b,CS1d
BC2,BC1,BC3,CS1c,BC4,CS1a,CS1d,CS1b
BC2,BC1,BC3,CS1c,BC4,CS1b,CS1a,CS1d
BC2,BC1,BC3,CS1c,BC4,CS1b,CS1d,CS1a
BC2,BC1,BC3,CS1c,BC4,CS1d,CS1a,CS1b
BC2,BC1,BC3,CS1c,BC4,CS1d,CS1b,CS1a
BC2,BC1,BC3,BC4,CS1a,CS1b,CS1c,CS1d
BC2,BC1,BC3,BC4,CS1a,CS1b,CS1d,CS1c
BC2,BC1,BC3,BC4,CS1a,CS1c,CS1b,CS1d
BC2,BC1,BC3,BC4,CS1a,CS1c,CS1d,CS1b
BC2,BC1,BC3,BC4,CS1a,CS1d,CS1b,CS1c
BC2,BC1,BC3,BC4,CS1a,CS1d,CS1c,CS1b

BC2,BC1,BC3,BC4,CS1b,CS1a,CS1c,CS1d
BC2,BC1,BC3,BC4,CS1b,CS1a,CS1d,CS1c
BC2,BC1,BC3,BC4,CS1b,CS1c,CS1a,CS1d
BC2,BC1,BC3,BC4,CS1b,CS1c,CS1d,CS1a
BC2,BC1,BC3,BC4,CS1b,CS1d,CS1a,CS1c
BC2,BC1,BC3,BC4,CS1b,CS1d,CS1c,CS1a
BC2,BC1,BC3,BC4,CS1c,CS1a,CS1b,CS1d
BC2,BC1,BC3,BC4,CS1c,CS1a,CS1d,CS1b
BC2,BC1,BC3,BC4,CS1c,CS1b,CS1a,CS1d
BC2,BC1,BC3,BC4,CS1c,CS1b,CS1d,CS1a
BC2,BC1,BC3,BC4,CS1c,CS1d,CS1a,CS1b
BC2,BC1,BC3,BC4,CS1c,CS1d,CS1b,CS1a
BC2,BC1,BC3,BC4,CS1d,CS1a,CS1b,CS1c
BC2,BC1,BC3,BC4,CS1d,CS1a,CS1c,CS1b
BC2,BC1,BC3,BC4,CS1d,CS1b,CS1a,CS1c
BC2,BC1,BC3,BC4,CS1d,CS1b,CS1c,CS1a
BC2,BC1,BC3,BC4,CS1d,CS1c,CS1a,CS1b
BC2,BC1,BC3,BC4,CS1d,CS1c,CS1b,CS1a
BC2,BC1,BC4,CS1a,CS1b,BC3,CS1c,CS1d
BC2,BC1,BC4,CS1a,CS1b,BC3,CS1d,CS1c
BC2,BC1,BC4,CS1a,CS1b,CS1d,BC3,CS1c
BC2,BC1,BC4,CS1a,BC3,CS1b,CS1c,CS1d
BC2,BC1,BC4,CS1a,BC3,CS1b,CS1d,CS1c
BC2,BC1,BC4,CS1a,BC3,CS1c,CS1b,CS1d
BC2,BC1,BC4,CS1a,BC3,CS1c,CS1d,CS1b
BC2,BC1,BC4,CS1a,BC3,CS1d,CS1b,CS1c
BC2,BC1,BC4,CS1a,BC3,CS1d,CS1c,CS1b
BC2,BC1,BC4,CS1a,CS1d,CS1b,BC3,CS1c
BC2,BC1,BC4,CS1a,CS1d,BC3,CS1b,CS1c
BC2,BC1,BC4,CS1a,CS1d,BC3,CS1c,CS1b
BC2,BC1,BC4,CS1b,CS1a,BC3,CS1c,CS1d
BC2,BC1,BC4,CS1b,CS1a,BC3,CS1d,CS1c
BC2,BC1,BC4,CS1b,CS1a,CS1d,BC3,CS1c
BC2,BC1,BC4,CS1b,BC3,CS1a,CS1c,CS1d
BC2,BC1,BC4,CS1b,BC3,CS1a,CS1d,CS1c
BC2,BC1,BC4,CS1b,BC3,CS1c,CS1a,CS1d
BC2,BC1,BC4,CS1b,BC3,CS1c,CS1d,CS1a
BC2,BC1,BC4,CS1b,BC3,CS1d,CS1a,CS1c
BC2,BC1,BC4,CS1b,BC3,CS1d,CS1c,CS1a
BC2,BC1,BC4,CS1b,CS1d,CS1a,BC3,CS1c
BC2,BC1,BC4,CS1b,CS1d,BC3,CS1a,CS1c
BC2,BC1,BC4,CS1b,CS1d,BC3,CS1c,CS1a
BC2,BC1,BC4,BC3,CS1a,CS1b,CS1c,CS1d
BC2,BC1,BC4,BC3,CS1a,CS1b,CS1d,CS1c
BC2,BC1,BC4,BC3,CS1a,CS1c,CS1b,CS1d
BC2,BC1,BC4,BC3,CS1a,CS1c,CS1d,CS1b

BC2,BC1,BC4,BC3,CS1a,CS1d,CS1b,CS1c
BC2,BC1,BC4,BC3,CS1a,CS1d,CS1c,CS1b
BC2,BC1,BC4,BC3,CS1b,CS1a,CS1c,CS1d
BC2,BC1,BC4,BC3,CS1b,CS1a,CS1d,CS1c
BC2,BC1,BC4,BC3,CS1b,CS1c,CS1a,CS1d
BC2,BC1,BC4,BC3,CS1b,CS1c,CS1d,CS1a
BC2,BC1,BC4,BC3,CS1b,CS1d,CS1a,CS1c
BC2,BC1,BC4,BC3,CS1b,CS1d,CS1c,CS1a
BC2,BC1,BC4,BC3,CS1c,CS1a,CS1b,CS1d
BC2,BC1,BC4,BC3,CS1c,CS1a,CS1d,CS1b
BC2,BC1,BC4,BC3,CS1c,CS1b,CS1a,CS1d
BC2,BC1,BC4,BC3,CS1c,CS1b,CS1d,CS1a
BC2,BC1,BC4,BC3,CS1c,CS1d,CS1a,CS1b
BC2,BC1,BC4,BC3,CS1c,CS1d,CS1b,CS1a
BC2,BC1,BC4,BC3,CS1d,CS1a,CS1b,CS1c
BC2,BC1,BC4,BC3,CS1d,CS1a,CS1c,CS1b
BC2,BC1,BC4,BC3,CS1d,CS1b,CS1a,CS1c
BC2,BC1,BC4,BC3,CS1d,CS1b,CS1c,CS1a
BC2,BC1,BC4,BC3,CS1d,CS1c,CS1a,CS1b
BC2,BC1,BC4,BC3,CS1d,CS1c,CS1b,CS1a
BC2,BC1,BC4,CS1d,CS1a,CS1b,BC3,CS1c
BC2,BC1,BC4,CS1d,CS1a,BC3,CS1b,CS1c
BC2,BC1,BC4,CS1d,CS1a,BC3,CS1c,CS1b
BC2,BC1,BC4,CS1d,CS1b,CS1a,BC3,CS1c
BC2,BC1,BC4,CS1d,CS1b,BC3,CS1a,CS1c
BC2,BC1,BC4,CS1d,CS1b,BC3,CS1c,CS1a
BC2,BC1,BC4,CS1d,BC3,CS1a,CS1b,CS1c
BC2,BC1,BC4,CS1d,BC3,CS1a,CS1c,CS1b
BC2,BC1,BC4,CS1d,BC3,CS1b,CS1a,CS1c
BC2,BC1,BC4,CS1d,BC3,CS1b,CS1c,CS1a
BC2,BC1,BC4,CS1d,BC3,CS1c,CS1a,CS1b
BC2,BC1,BC4,CS1d,BC3,CS1c,CS1b,CS1a
BC2,CS1b,BC1,CS1a,BC3,CS1c,BC4,CS1d
BC2,CS1b,BC1,CS1a,BC3,BC4,CS1c,CS1d
BC2,CS1b,BC1,CS1a,BC3,BC4,CS1d,CS1c
BC2,CS1b,BC1,CS1a,BC4,BC3,CS1c,CS1d
BC2,CS1b,BC1,CS1a,BC4,BC3,CS1d,CS1c
BC2,CS1b,BC1,CS1a,BC4,CS1d,BC3,CS1c
BC2,CS1b,BC1,BC3,CS1a,CS1c,BC4,CS1d
BC2,CS1b,BC1,BC3,CS1a,BC4,CS1c,CS1d
BC2,CS1b,BC1,BC3,CS1a,BC4,CS1d,CS1c
BC2,CS1b,BC1,BC3,CS1c,CS1a,BC4,CS1d
BC2,CS1b,BC1,BC3,CS1c,BC4,CS1a,CS1d
BC2,CS1b,BC1,BC3,CS1c,BC4,CS1d,CS1a
BC2,CS1b,BC1,BC3,BC4,CS1a,CS1c,CS1d
BC2,CS1b,BC1,BC3,BC4,CS1a,CS1d,CS1c

BC2,CS1b,BC1,BC3,BC4,CS1c,CS1a,CS1d
BC2,CS1b,BC1,BC3,BC4,CS1c,CS1d,CS1a
BC2,CS1b,BC1,BC3,BC4,CS1d,CS1a,CS1c
BC2,CS1b,BC1,BC3,BC4,CS1d,CS1c,CS1a
BC2,CS1b,BC1,BC4,CS1a,BC3,CS1c,CS1d
BC2,CS1b,BC1,BC4,CS1a,BC3,CS1d,CS1c
BC2,CS1b,BC1,BC4,CS1a,CS1d,BC3,CS1c
BC2,CS1b,BC1,BC4,BC3,CS1a,CS1c,CS1d
BC2,CS1b,BC1,BC4,BC3,CS1a,CS1d,CS1c
BC2,CS1b,BC1,BC4,BC3,CS1c,CS1a,CS1d
BC2,CS1b,BC1,BC4,BC3,CS1c,CS1d,CS1a
BC2,CS1b,BC1,BC4,BC3,CS1d,CS1a,CS1c
BC2,CS1b,BC1,BC4,BC3,CS1d,CS1c,CS1a
BC2,CS1b,BC1,BC4,CS1d,CS1a,BC3,CS1c
BC2,CS1b,BC1,BC4,CS1d,BC3,CS1a,CS1c
BC2,CS1b,BC1,BC4,CS1d,BC3,CS1c,CS1a
BC2,CS1b,BC3,BC1,CS1a,CS1c,BC4,CS1d
BC2,CS1b,BC3,BC1,CS1a,BC4,CS1c,CS1d
BC2,CS1b,BC3,BC1,CS1a,BC4,CS1d,CS1c
BC2,CS1b,BC3,BC1,CS1c,CS1a,BC4,CS1d
BC2,CS1b,BC3,BC1,CS1c,BC4,CS1a,CS1d
BC2,CS1b,BC3,BC1,CS1c,BC4,CS1d,CS1a
BC2,CS1b,BC3,BC1,BC4,CS1a,CS1c,CS1d
BC2,CS1b,BC3,BC1,BC4,CS1a,CS1d,CS1c
BC2,CS1b,BC3,BC1,BC4,CS1c,CS1a,CS1d
BC2,CS1b,BC3,BC1,BC4,CS1c,CS1d,CS1a
BC2,CS1b,BC3,BC1,BC4,CS1d,CS1a,CS1c
BC2,CS1b,BC3,BC1,BC4,CS1d,CS1c,CS1a
BC2,CS1b,BC3,CS1c,BC1,CS1a,BC4,CS1d
BC2,CS1b,BC3,CS1c,BC1,BC4,CS1a,CS1d
BC2,CS1b,BC3,CS1c,BC1,BC4,CS1d,CS1a
BC2,CS1b,BC3,CS1c,BC4,BC1,CS1a,CS1d
BC2,CS1b,BC3,CS1c,BC4,BC1,CS1d,CS1a
BC2,CS1b,BC3,CS1c,BC4,CS1d,BC1,CS1a
BC2,CS1b,BC3,BC4,BC1,CS1a,CS1c,CS1d
BC2,CS1b,BC3,BC4,BC1,CS1a,CS1d,CS1c
BC2,CS1b,BC3,BC4,BC1,CS1c,CS1a,CS1d
BC2,CS1b,BC3,BC4,BC1,CS1c,CS1d,CS1a
BC2,CS1b,BC3,BC4,BC1,CS1d,CS1a,CS1c
BC2,CS1b,BC3,BC4,BC1,CS1d,CS1c,CS1a
BC2,CS1b,BC3,BC4,CS1c,BC1,CS1a,CS1d
BC2,CS1b,BC3,BC4,CS1c,BC1,CS1d,CS1a
BC2,CS1b,BC3,BC4,CS1c,CS1d,BC1,CS1a
BC2,CS1b,BC3,BC4,CS1d,BC1,CS1a,CS1c
BC2,CS1b,BC3,BC4,CS1d,BC1,CS1c,CS1a
BC2,CS1b,BC3,BC4,CS1d,CS1c,BC1,CS1a

BC2,CS1b,BC4,BC1,CS1a,BC3,CS1c,CS1d
BC2,CS1b,BC4,BC1,CS1a,BC3,CS1d,CS1c
BC2,CS1b,BC4,BC1,CS1a,CS1d,BC3,CS1c
BC2,CS1b,BC4,BC1,BC3,CS1a,CS1c,CS1d
BC2,CS1b,BC4,BC1,BC3,CS1a,CS1d,CS1c
BC2,CS1b,BC4,BC1,BC3,CS1c,CS1a,CS1d
BC2,CS1b,BC4,BC1,BC3,CS1c,CS1d,CS1a
BC2,CS1b,BC4,BC1,BC3,CS1d,CS1a,CS1c
BC2,CS1b,BC4,BC1,BC3,CS1d,CS1c,CS1a
BC2,CS1b,BC4,BC1,CS1d,CS1a,BC3,CS1c
BC2,CS1b,BC4,BC1,CS1d,BC3,CS1a,CS1c
BC2,CS1b,BC4,BC1,CS1d,BC3,CS1c,CS1a
BC2,CS1b,BC4,BC3,BC1,CS1a,CS1c,CS1d
BC2,CS1b,BC4,BC3,BC1,CS1a,CS1d,CS1c
BC2,CS1b,BC4,BC3,BC1,CS1c,CS1a,CS1d
BC2,CS1b,BC4,BC3,BC1,CS1c,CS1d,CS1a
BC2,CS1b,BC4,BC3,BC1,CS1d,CS1a,CS1c
BC2,CS1b,BC4,BC3,BC1,CS1d,CS1c,CS1a
BC2,CS1b,BC4,BC3,CS1c,BC1,CS1a,CS1d
BC2,CS1b,BC4,BC3,CS1c,BC1,CS1d,CS1a
BC2,CS1b,BC4,BC3,CS1c,CS1d,BC1,CS1a
BC2,CS1b,BC4,BC3,CS1d,BC1,CS1a,CS1c
BC2,CS1b,BC4,BC3,CS1d,BC1,CS1c,CS1a
BC2,CS1b,BC4,BC3,CS1d,CS1c,BC1,CS1a
BC2,CS1b,BC4,CS1d,BC1,CS1a,BC3,CS1c
BC2,CS1b,BC4,CS1d,BC1,BC3,CS1a,CS1c
BC2,CS1b,BC4,CS1d,BC1,BC3,CS1c,CS1a
BC2,CS1b,BC4,CS1d,BC3,BC1,CS1a,CS1c
BC2,CS1b,BC4,CS1d,BC3,BC1,CS1c,CS1a
BC2,CS1b,BC4,CS1d,BC3,CS1c,BC1,CS1a
BC2,BC3,BC1,CS1a,CS1b,CS1c,BC4,CS1d
BC2,BC3,BC1,CS1a,CS1b,BC4,CS1c,CS1d
BC2,BC3,BC1,CS1a,CS1b,BC4,CS1d,CS1c
BC2,BC3,BC1,CS1a,CS1c,CS1b,BC4,CS1d
BC2,BC3,BC1,CS1a,CS1c,BC4,CS1b,CS1d
BC2,BC3,BC1,CS1a,CS1c,BC4,CS1d,CS1b
BC2,BC3,BC1,CS1a,BC4,CS1b,CS1c,CS1d
BC2,BC3,BC1,CS1a,BC4,CS1b,CS1d,CS1c
BC2,BC3,BC1,CS1a,BC4,CS1c,CS1b,CS1d
BC2,BC3,BC1,CS1a,BC4,CS1c,CS1d,CS1b
BC2,BC3,BC1,CS1a,BC4,CS1d,CS1b,CS1c
BC2,BC3,BC1,CS1a,BC4,CS1d,CS1c,CS1b
BC2,BC3,BC1,CS1b,CS1a,CS1c,BC4,CS1d
BC2,BC3,BC1,CS1b,CS1a,BC4,CS1c,CS1d
BC2,BC3,BC1,CS1b,CS1a,BC4,CS1d,CS1c
BC2,BC3,BC1,CS1b,CS1c,CS1a,BC4,CS1d

BC2,BC3,BC1,CS1b,CS1c,BC4,CS1a,CS1d
BC2,BC3,BC1,CS1b,CS1c,BC4,CS1d,CS1a
BC2,BC3,BC1,CS1b,BC4,CS1a,CS1c,CS1d
BC2,BC3,BC1,CS1b,BC4,CS1a,CS1d,CS1c
BC2,BC3,BC1,CS1b,BC4,CS1c,CS1a,CS1d
BC2,BC3,BC1,CS1b,BC4,CS1c,CS1d,CS1a
BC2,BC3,BC1,CS1b,BC4,CS1d,CS1a,CS1c
BC2,BC3,BC1,CS1b,BC4,CS1d,CS1c,CS1a
BC2,BC3,BC1,CS1c,CS1a,CS1b,BC4,CS1d
BC2,BC3,BC1,CS1c,CS1a,BC4,CS1b,CS1d
BC2,BC3,BC1,CS1c,CS1a,BC4,CS1d,CS1b
BC2,BC3,BC1,CS1c,CS1b,CS1a,BC4,CS1d
BC2,BC3,BC1,CS1c,CS1b,BC4,CS1a,CS1d
BC2,BC3,BC1,CS1c,CS1b,BC4,CS1d,CS1a
BC2,BC3,BC1,CS1c,BC4,CS1a,CS1b,CS1d
BC2,BC3,BC1,CS1c,BC4,CS1a,CS1d,CS1b
BC2,BC3,BC1,CS1c,BC4,CS1b,CS1a,CS1d
BC2,BC3,BC1,CS1c,BC4,CS1b,CS1d,CS1a
BC2,BC3,BC1,CS1c,BC4,CS1d,CS1a,CS1b
BC2,BC3,BC1,CS1c,BC4,CS1d,CS1b,CS1a
BC2,BC3,BC1,BC4,CS1a,CS1b,CS1c,CS1d
BC2,BC3,BC1,BC4,CS1a,CS1b,CS1d,CS1c
BC2,BC3,BC1,BC4,CS1a,CS1c,CS1b,CS1d
BC2,BC3,BC1,BC4,CS1a,CS1c,CS1d,CS1b
BC2,BC3,BC1,BC4,CS1a,CS1d,CS1b,CS1c
BC2,BC3,BC1,BC4,CS1a,CS1d,CS1c,CS1b
BC2,BC3,BC1,BC4,CS1b,CS1a,CS1c,CS1d
BC2,BC3,BC1,BC4,CS1b,CS1a,CS1d,CS1c
BC2,BC3,BC1,BC4,CS1b,CS1c,CS1a,CS1d
BC2,BC3,BC1,BC4,CS1b,CS1c,CS1d,CS1a
BC2,BC3,BC1,BC4,CS1b,CS1d,CS1a,CS1c
BC2,BC3,BC1,BC4,CS1b,CS1d,CS1c,CS1a
BC2,BC3,BC1,BC4,CS1c,CS1a,CS1b,CS1d
BC2,BC3,BC1,BC4,CS1c,CS1a,CS1d,CS1b
BC2,BC3,BC1,BC4,CS1c,CS1b,CS1a,CS1d
BC2,BC3,BC1,BC4,CS1c,CS1b,CS1d,CS1a
BC2,BC3,BC1,BC4,CS1c,CS1d,CS1a,CS1b
BC2,BC3,BC1,BC4,CS1c,CS1d,CS1b,CS1a
BC2,BC3,BC1,BC4,CS1d,CS1a,CS1b,CS1c
BC2,BC3,BC1,BC4,CS1d,CS1a,CS1c,CS1b
BC2,BC3,BC1,BC4,CS1d,CS1b,CS1a,CS1c
BC2,BC3,BC1,BC4,CS1d,CS1b,CS1c,CS1a
BC2,BC3,BC1,BC4,CS1d,CS1c,CS1a,CS1b
BC2,BC3,BC1,BC4,CS1d,CS1c,CS1b,CS1a
BC2,BC3,CS1b,BC1,CS1a,CS1c,BC4,CS1d
BC2,BC3,CS1b,BC1,CS1a,BC4,CS1c,CS1d

BC2,BC3,CS1b,BC1,CS1a,BC4,CS1d,CS1c
BC2,BC3,CS1b,BC1,CS1c,CS1a,BC4,CS1d
BC2,BC3,CS1b,BC1,CS1c,BC4,CS1a,CS1d
BC2,BC3,CS1b,BC1,CS1c,BC4,CS1d,CS1a
BC2,BC3,CS1b,BC1,BC4,CS1a,CS1c,CS1d
BC2,BC3,CS1b,BC1,BC4,CS1a,CS1d,CS1c
BC2,BC3,CS1b,BC1,BC4,CS1c,CS1a,CS1d
BC2,BC3,CS1b,BC1,BC4,CS1c,CS1d,CS1a
BC2,BC3,CS1b,BC1,BC4,CS1d,CS1a,CS1c
BC2,BC3,CS1b,BC1,BC4,CS1d,CS1c,CS1a
BC2,BC3,CS1b,CS1c,BC1,CS1a,BC4,CS1d
BC2,BC3,CS1b,CS1c,BC1,BC4,CS1a,CS1d
BC2,BC3,CS1b,CS1c,BC1,BC4,CS1d,CS1a
BC2,BC3,CS1b,CS1c,BC4,BC1,CS1a,CS1d
BC2,BC3,CS1b,CS1c,BC4,BC1,CS1d,CS1a
BC2,BC3,CS1b,CS1c,BC4,CS1d,BC1,CS1a
BC2,BC3,CS1b,BC4,BC1,CS1a,CS1c,CS1d
BC2,BC3,CS1b,BC4,BC1,CS1a,CS1d,CS1c
BC2,BC3,CS1b,BC4,BC1,CS1c,CS1a,CS1d
BC2,BC3,CS1b,BC4,BC1,CS1c,CS1d,CS1a
BC2,BC3,CS1b,BC4,BC1,CS1d,CS1a,CS1c
BC2,BC3,CS1b,BC4,BC1,CS1d,CS1c,CS1a
BC2,BC3,CS1b,BC4,CS1c,BC1,CS1a,CS1d
BC2,BC3,CS1b,BC4,CS1c,BC1,CS1d,CS1a
BC2,BC3,CS1b,BC4,CS1c,CS1d,BC1,CS1a
BC2,BC3,CS1b,BC4,CS1d,BC1,CS1a,CS1c
BC2,BC3,CS1b,BC4,CS1d,BC1,CS1c,CS1a
BC2,BC3,CS1b,BC4,CS1d,CS1c,BC1,CS1a
BC2,BC3,CS1c,BC1,CS1a,CS1b,BC4,CS1d
BC2,BC3,CS1c,BC1,CS1a,BC4,CS1b,CS1d
BC2,BC3,CS1c,BC1,CS1a,BC4,CS1d,CS1b
BC2,BC3,CS1c,BC1,CS1b,CS1a,BC4,CS1d
BC2,BC3,CS1c,BC1,CS1b,BC4,CS1a,CS1d
BC2,BC3,CS1c,BC1,CS1b,BC4,CS1d,CS1a
BC2,BC3,CS1c,BC1,BC4,CS1a,CS1b,CS1d
BC2,BC3,CS1c,BC1,BC4,CS1a,CS1d,CS1b
BC2,BC3,CS1c,BC1,BC4,CS1b,CS1a,CS1d
BC2,BC3,CS1c,BC1,BC4,CS1b,CS1d,CS1a
BC2,BC3,CS1c,BC1,BC4,CS1d,CS1a,CS1b
BC2,BC3,CS1c,BC1,BC4,CS1d,CS1b,CS1a
BC2,BC3,CS1c,CS1b,BC1,CS1a,BC4,CS1d
BC2,BC3,CS1c,CS1b,BC1,BC4,CS1a,CS1d
BC2,BC3,CS1c,CS1b,BC1,BC4,CS1d,CS1a
BC2,BC3,CS1c,CS1b,BC4,BC1,CS1a,CS1d
BC2,BC3,CS1c,CS1b,BC4,BC1,CS1d,CS1a
BC2,BC3,CS1c,CS1b,BC4,CS1d,BC1,CS1a

BC2,BC3,CS1c,BC4,BC1,CS1a,CS1b,CS1d
BC2,BC3,CS1c,BC4,BC1,CS1a,CS1d,CS1b
BC2,BC3,CS1c,BC4,BC1,CS1b,CS1a,CS1d
BC2,BC3,CS1c,BC4,BC1,CS1b,CS1d,CS1a
BC2,BC3,CS1c,BC4,BC1,CS1d,CS1a,CS1b
BC2,BC3,CS1c,BC4,BC1,CS1d,CS1b,CS1a
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1a,CS1d
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1d,CS1a
BC2,BC3,CS1c,BC4,CS1b,CS1d,BC1,CS1a
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1a,CS1b
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1b,CS1a
BC2,BC3,CS1c,BC4,CS1d,CS1b,BC1,CS1a
BC2,BC3,BC4,BC1,CS1a,CS1b,CS1c,CS1d
BC2,BC3,BC4,BC1,CS1a,CS1b,CS1d,CS1c
BC2,BC3,BC4,BC1,CS1a,CS1c,CS1b,CS1d
BC2,BC3,BC4,BC1,CS1a,CS1c,CS1d,CS1b
BC2,BC3,BC4,BC1,CS1a,CS1d,CS1b,CS1c
BC2,BC3,BC4,BC1,CS1a,CS1d,CS1c,CS1b
BC2,BC3,BC4,BC1,CS1b,CS1a,CS1c,CS1d
BC2,BC3,BC4,BC1,CS1b,CS1a,CS1d,CS1c
BC2,BC3,BC4,BC1,CS1b,CS1c,CS1a,CS1d
BC2,BC3,BC4,BC1,CS1b,CS1c,CS1d,CS1a
BC2,BC3,BC4,BC1,CS1b,CS1d,CS1a,CS1c
BC2,BC3,BC4,BC1,CS1b,CS1d,CS1c,CS1a
BC2,BC3,BC4,BC1,CS1c,CS1a,CS1b,CS1d
BC2,BC3,BC4,BC1,CS1c,CS1a,CS1d,CS1b
BC2,BC3,BC4,BC1,CS1c,CS1b,CS1a,CS1d
BC2,BC3,BC4,BC1,CS1c,CS1b,CS1d,CS1a
BC2,BC3,BC4,BC1,CS1c,CS1d,CS1a,CS1b
BC2,BC3,BC4,BC1,CS1c,CS1d,CS1b,CS1a
BC2,BC3,BC4,BC1,CS1d,CS1a,CS1b,CS1c
BC2,BC3,BC4,BC1,CS1d,CS1a,CS1c,CS1b
BC2,BC3,BC4,BC1,CS1d,CS1b,CS1a,CS1c
BC2,BC3,BC4,BC1,CS1d,CS1b,CS1c,CS1a
BC2,BC3,BC4,BC1,CS1d,CS1c,CS1a,CS1b
BC2,BC3,BC4,BC1,CS1d,CS1c,CS1b,CS1a
BC2,BC3,BC4,CS1b,BC1,CS1a,CS1c,CS1d
BC2,BC3,BC4,CS1b,BC1,CS1a,CS1d,CS1c
BC2,BC3,BC4,CS1b,BC1,CS1c,CS1a,CS1d
BC2,BC3,BC4,CS1b,BC1,CS1c,CS1d,CS1a
BC2,BC3,BC4,CS1b,BC1,CS1d,CS1a,CS1c
BC2,BC3,BC4,CS1b,BC1,CS1d,CS1c,CS1a
BC2,BC3,BC4,CS1b,CS1c,BC1,CS1a,CS1d
BC2,BC3,BC4,CS1b,CS1c,BC1,CS1d,CS1a
BC2,BC3,BC4,CS1b,CS1c,CS1d,BC1,CS1a
BC2,BC3,BC4,CS1b,CS1d,BC1,CS1a,CS1c

BC2,BC3,BC4,CS1b,CS1d,BC1,CS1c,CS1a
BC2,BC3,BC4,CS1b,CS1d,CS1c,BC1,CS1a
BC2,BC3,BC4,CS1c,BC1,CS1a,CS1b,CS1d
BC2,BC3,BC4,CS1c,BC1,CS1a,CS1d,CS1b
BC2,BC3,BC4,CS1c,BC1,CS1b,CS1a,CS1d
BC2,BC3,BC4,CS1c,BC1,CS1b,CS1d,CS1a
BC2,BC3,BC4,CS1c,BC1,CS1d,CS1a,CS1b
BC2,BC3,BC4,CS1c,BC1,CS1d,CS1b,CS1a
BC2,BC3,BC4,CS1c,CS1b,BC1,CS1a,CS1d
BC2,BC3,BC4,CS1c,CS1b,BC1,CS1d,CS1a
BC2,BC3,BC4,CS1c,CS1b,CS1d,BC1,CS1a
BC2,BC3,BC4,CS1c,CS1d,BC1,CS1a,CS1b
BC2,BC3,BC4,CS1c,CS1d,BC1,CS1b,CS1a
BC2,BC3,BC4,CS1c,CS1d,CS1b,BC1,CS1a
BC2,BC3,BC4,CS1d,BC1,CS1a,CS1b,CS1c
BC2,BC3,BC4,CS1d,BC1,CS1a,CS1c,CS1b
BC2,BC3,BC4,CS1d,BC1,CS1b,CS1a,CS1c
BC2,BC3,BC4,CS1d,BC1,CS1b,CS1c,CS1a
BC2,BC3,BC4,CS1d,BC1,CS1c,CS1a,CS1b
BC2,BC3,BC4,CS1d,BC1,CS1c,CS1b,CS1a
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1a,CS1c
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1c,CS1a
BC2,BC3,BC4,CS1d,CS1b,CS1c,BC1,CS1a
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1a,CS1b
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1b,CS1a
BC2,BC3,BC4,CS1d,CS1c,CS1b,BC1,CS1a
BC2,BC4,BC1,CS1a,CS1b,BC3,CS1c,CS1d
BC2,BC4,BC1,CS1a,CS1b,BC3,CS1d,CS1c
BC2,BC4,BC1,CS1a,CS1b,CS1d,BC3,CS1c
BC2,BC4,BC1,CS1a,BC3,CS1b,CS1c,CS1d
BC2,BC4,BC1,CS1a,BC3,CS1b,CS1d,CS1c
BC2,BC4,BC1,CS1a,BC3,CS1c,CS1b,CS1d
BC2,BC4,BC1,CS1a,BC3,CS1c,CS1d,CS1b
BC2,BC4,BC1,CS1a,BC3,CS1d,CS1b,CS1c
BC2,BC4,BC1,CS1a,BC3,CS1d,CS1c,CS1b
BC2,BC4,BC1,CS1a,CS1d,CS1b,BC3,CS1c
BC2,BC4,BC1,CS1a,CS1d,BC3,CS1b,CS1c
BC2,BC4,BC1,CS1a,CS1d,BC3,CS1c,CS1b
BC2,BC4,BC1,CS1b,CS1a,BC3,CS1c,CS1d
BC2,BC4,BC1,CS1b,CS1a,BC3,CS1d,CS1c
BC2,BC4,BC1,CS1b,CS1a,CS1d,BC3,CS1c
BC2,BC4,BC1,CS1b,BC3,CS1a,CS1c,CS1d
BC2,BC4,BC1,CS1b,BC3,CS1a,CS1d,CS1c
BC2,BC4,BC1,CS1b,BC3,CS1c,CS1a,CS1d
BC2,BC4,BC1,CS1b,BC3,CS1c,CS1d,CS1a
BC2,BC4,BC1,CS1b,BC3,CS1d,CS1a,CS1c

BC2,BC4,BC1,CS1b,BC3,CS1d,CS1c,CS1a
BC2,BC4,BC1,CS1b,CS1d,CS1a,BC3,CS1c
BC2,BC4,BC1,CS1b,CS1d,BC3,CS1a,CS1c
BC2,BC4,BC1,CS1b,CS1d,BC3,CS1c,CS1a
BC2,BC4,BC1,BC3,CS1a,CS1b,CS1c,CS1d
BC2,BC4,BC1,BC3,CS1a,CS1b,CS1d,CS1c
BC2,BC4,BC1,BC3,CS1a,CS1c,CS1b,CS1d
BC2,BC4,BC1,BC3,CS1a,CS1c,CS1d,CS1b
BC2,BC4,BC1,BC3,CS1a,CS1d,CS1b,CS1c
BC2,BC4,BC1,BC3,CS1a,CS1d,CS1c,CS1b
BC2,BC4,BC1,BC3,CS1b,CS1a,CS1c,CS1d
BC2,BC4,BC1,BC3,CS1b,CS1a,CS1d,CS1c
BC2,BC4,BC1,BC3,CS1b,CS1c,CS1a,CS1d
BC2,BC4,BC1,BC3,CS1b,CS1c,CS1d,CS1a
BC2,BC4,BC1,BC3,CS1b,CS1d,CS1a,CS1c
BC2,BC4,BC1,BC3,CS1b,CS1d,CS1c,CS1a
BC2,BC4,BC1,BC3,CS1c,CS1a,CS1b,CS1d
BC2,BC4,BC1,BC3,CS1c,CS1a,CS1d,CS1b
BC2,BC4,BC1,BC3,CS1c,CS1b,CS1a,CS1d
BC2,BC4,BC1,BC3,CS1c,CS1b,CS1d,CS1a
BC2,BC4,BC1,BC3,CS1c,CS1d,CS1a,CS1b
BC2,BC4,BC1,BC3,CS1c,CS1d,CS1b,CS1a
BC2,BC4,BC1,BC3,CS1d,CS1a,CS1b,CS1c
BC2,BC4,BC1,BC3,CS1d,CS1a,CS1c,CS1b
BC2,BC4,BC1,BC3,CS1d,CS1b,CS1a,CS1c
BC2,BC4,BC1,BC3,CS1d,CS1b,CS1c,CS1a
BC2,BC4,BC1,BC3,CS1d,CS1c,CS1a,CS1b
BC2,BC4,BC1,BC3,CS1d,CS1c,CS1b,CS1a
BC2,BC4,BC1,CS1d,CS1a,CS1b,BC3,CS1c
BC2,BC4,BC1,CS1d,CS1a,BC3,CS1b,CS1c
BC2,BC4,BC1,CS1d,CS1a,BC3,CS1c,CS1b
BC2,BC4,BC1,CS1d,CS1b,CS1a,BC3,CS1c
BC2,BC4,BC1,CS1d,CS1b,BC3,CS1a,CS1c
BC2,BC4,BC1,CS1d,CS1b,BC3,CS1c,CS1a
BC2,BC4,BC1,CS1d,BC3,CS1a,CS1b,CS1c
BC2,BC4,BC1,CS1d,BC3,CS1a,CS1c,CS1b
BC2,BC4,BC1,CS1d,BC3,CS1b,CS1a,CS1c
BC2,BC4,BC1,CS1d,BC3,CS1b,CS1c,CS1a
BC2,BC4,BC1,CS1d,BC3,CS1c,CS1a,CS1b
BC2,BC4,BC1,CS1d,BC3,CS1c,CS1b,CS1a
BC2,BC4,CS1b,BC1,CS1a,BC3,CS1c,CS1d
BC2,BC4,CS1b,BC1,CS1a,BC3,CS1d,CS1c
BC2,BC4,CS1b,BC1,CS1a,CS1d,BC3,CS1c
BC2,BC4,CS1b,BC1,BC3,CS1a,CS1c,CS1d
BC2,BC4,CS1b,BC1,BC3,CS1a,CS1d,CS1c
BC2,BC4,CS1b,BC1,BC3,CS1c,CS1a,CS1d

BC2,BC4,CS1b,BC1,BC3,CS1c,CS1d,CS1a
BC2,BC4,CS1b,BC1,BC3,CS1d,CS1a,CS1c
BC2,BC4,CS1b,BC1,BC3,CS1d,CS1c,CS1a
BC2,BC4,CS1b,BC1,CS1d,CS1a,BC3,CS1c
BC2,BC4,CS1b,BC1,CS1d,BC3,CS1a,CS1c
BC2,BC4,CS1b,BC1,CS1d,BC3,CS1c,CS1a
BC2,BC4,CS1b,BC3,BC1,CS1a,CS1c,CS1d
BC2,BC4,CS1b,BC3,BC1,CS1a,CS1d,CS1c
BC2,BC4,CS1b,BC3,BC1,CS1c,CS1a,CS1d
BC2,BC4,CS1b,BC3,BC1,CS1c,CS1d,CS1a
BC2,BC4,CS1b,BC3,BC1,CS1d,CS1a,CS1c
BC2,BC4,CS1b,BC3,BC1,CS1d,CS1c,CS1a
BC2,BC4,CS1b,BC3,CS1c,BC1,CS1a,CS1d
BC2,BC4,CS1b,BC3,CS1c,BC1,CS1d,CS1a
BC2,BC4,CS1b,BC3,CS1c,CS1d,BC1,CS1a
BC2,BC4,CS1b,BC3,CS1d,BC1,CS1a,CS1c
BC2,BC4,CS1b,BC3,CS1d,BC1,CS1c,CS1a
BC2,BC4,CS1b,BC3,CS1d,CS1c,BC1,CS1a
BC2,BC4,CS1b,CS1d,BC1,CS1a,BC3,CS1c
BC2,BC4,CS1b,CS1d,BC1,BC3,CS1a,CS1c
BC2,BC4,CS1b,CS1d,BC1,BC3,CS1c,CS1a
BC2,BC4,CS1b,CS1d,BC3,BC1,CS1a,CS1c
BC2,BC4,CS1b,CS1d,BC3,BC1,CS1c,CS1a
BC2,BC4,CS1b,CS1d,BC3,CS1c,BC1,CS1a
BC2,BC4,BC3,BC1,CS1a,CS1b,CS1c,CS1d
BC2,BC4,BC3,BC1,CS1a,CS1b,CS1d,CS1c
BC2,BC4,BC3,BC1,CS1a,CS1c,CS1b,CS1d
BC2,BC4,BC3,BC1,CS1a,CS1c,CS1d,CS1b
BC2,BC4,BC3,BC1,CS1a,CS1d,CS1b,CS1c
BC2,BC4,BC3,BC1,CS1a,CS1d,CS1c,CS1b
BC2,BC4,BC3,BC1,CS1b,CS1a,CS1c,CS1d
BC2,BC4,BC3,BC1,CS1b,CS1a,CS1d,CS1c
BC2,BC4,BC3,BC1,CS1b,CS1c,CS1a,CS1d
BC2,BC4,BC3,BC1,CS1b,CS1c,CS1d,CS1a
BC2,BC4,BC3,BC1,CS1b,CS1d,CS1a,CS1c
BC2,BC4,BC3,BC1,CS1b,CS1d,CS1c,CS1a
BC2,BC4,BC3,BC1,CS1c,CS1a,CS1b,CS1d
BC2,BC4,BC3,BC1,CS1c,CS1a,CS1d,CS1b
BC2,BC4,BC3,BC1,CS1c,CS1b,CS1a,CS1d
BC2,BC4,BC3,BC1,CS1c,CS1b,CS1d,CS1a
BC2,BC4,BC3,BC1,CS1c,CS1d,CS1a,CS1b
BC2,BC4,BC3,BC1,CS1c,CS1d,CS1b,CS1a
BC2,BC4,BC3,BC1,CS1d,CS1a,CS1b,CS1c
BC2,BC4,BC3,BC1,CS1d,CS1a,CS1c,CS1b
BC2,BC4,BC3,BC1,CS1d,CS1b,CS1a,CS1c
BC2,BC4,BC3,BC1,CS1d,CS1b,CS1c,CS1a

BC2,BC4,BC3,BC1,CS1d,CS1c,CS1a,CS1b
BC2,BC4,BC3,BC1,CS1d,CS1c,CS1b,CS1a
BC2,BC4,BC3,CS1b,BC1,CS1a,CS1c,CS1d
BC2,BC4,BC3,CS1b,BC1,CS1a,CS1d,CS1c
BC2,BC4,BC3,CS1b,BC1,CS1c,CS1a,CS1d
BC2,BC4,BC3,CS1b,BC1,CS1c,CS1d,CS1a
BC2,BC4,BC3,CS1b,BC1,CS1d,CS1a,CS1c
BC2,BC4,BC3,CS1b,BC1,CS1d,CS1c,CS1a
BC2,BC4,BC3,CS1b,CS1c,BC1,CS1a,CS1d
BC2,BC4,BC3,CS1b,CS1c,BC1,CS1d,CS1a
BC2,BC4,BC3,CS1b,CS1c,CS1d,BC1,CS1a
BC2,BC4,BC3,CS1b,CS1d,BC1,CS1a,CS1c
BC2,BC4,BC3,CS1b,CS1d,BC1,CS1c,CS1a
BC2,BC4,BC3,CS1b,CS1d,CS1c,BC1,CS1a
BC2,BC4,BC3,CS1c,BC1,CS1a,CS1b,CS1d
BC2,BC4,BC3,CS1c,BC1,CS1a,CS1d,CS1b
BC2,BC4,BC3,CS1c,BC1,CS1b,CS1a,CS1d
BC2,BC4,BC3,CS1c,BC1,CS1b,CS1d,CS1a
BC2,BC4,BC3,CS1c,BC1,CS1d,CS1a,CS1b
BC2,BC4,BC3,CS1c,BC1,CS1d,CS1b,CS1a
BC2,BC4,BC3,CS1c,CS1b,BC1,CS1a,CS1d
BC2,BC4,BC3,CS1c,CS1b,BC1,CS1d,CS1a
BC2,BC4,BC3,CS1c,CS1b,CS1d,BC1,CS1a
BC2,BC4,BC3,CS1c,CS1d,BC1,CS1a,CS1b
BC2,BC4,BC3,CS1c,CS1d,BC1,CS1b,CS1a
BC2,BC4,BC3,CS1c,CS1d,CS1b,BC1,CS1a
BC2,BC4,BC3,CS1d,BC1,CS1a,CS1b,CS1c
BC2,BC4,BC3,CS1d,BC1,CS1a,CS1c,CS1b
BC2,BC4,BC3,CS1d,BC1,CS1b,CS1a,CS1c
BC2,BC4,BC3,CS1d,BC1,CS1b,CS1c,CS1a
BC2,BC4,BC3,CS1d,BC1,CS1c,CS1a,CS1b
BC2,BC4,BC3,CS1d,BC1,CS1c,CS1b,CS1a
BC2,BC4,BC3,CS1d,CS1b,BC1,CS1a,CS1c
BC2,BC4,BC3,CS1d,CS1b,BC1,CS1c,CS1a
BC2,BC4,BC3,CS1d,CS1b,CS1c,BC1,CS1a
BC2,BC4,BC3,CS1d,CS1c,BC1,CS1a,CS1b
BC2,BC4,BC3,CS1d,CS1c,BC1,CS1b,CS1a
BC2,BC4,BC3,CS1d,CS1c,CS1b,BC1,CS1a
BC2,BC4,CS1d,BC1,CS1a,CS1b,BC3,CS1c
BC2,BC4,CS1d,BC1,CS1a,BC3,CS1b,CS1c
BC2,BC4,CS1d,BC1,CS1a,BC3,CS1c,CS1b
BC2,BC4,CS1d,BC1,CS1b,CS1a,BC3,CS1c
BC2,BC4,CS1d,BC1,CS1b,BC3,CS1a,CS1c
BC2,BC4,CS1d,BC1,CS1b,BC3,CS1c,CS1a
BC2,BC4,CS1d,BC1,BC3,CS1a,CS1b,CS1c
BC2,BC4,CS1d,BC1,BC3,CS1a,CS1c,CS1b

BC2,BC4,CS1d,BC1,BC3,CS1b,CS1a,CS1c
BC2,BC4,CS1d,BC1,BC3,CS1b,CS1c,CS1a
BC2,BC4,CS1d,BC1,BC3,CS1c,CS1a,CS1b
BC2,BC4,CS1d,BC1,BC3,CS1c,CS1b,CS1a
BC2,BC4,CS1d,CS1b,BC1,CS1a,BC3,CS1c
BC2,BC4,CS1d,CS1b,BC1,BC3,CS1a,CS1c
BC2,BC4,CS1d,CS1b,BC1,BC3,CS1c,CS1a
BC2,BC4,CS1d,CS1b,BC3,BC1,CS1a,CS1c
BC2,BC4,CS1d,CS1b,BC3,BC1,CS1c,CS1a
BC2,BC4,CS1d,CS1b,BC3,CS1c,BC1,CS1a
BC2,BC4,CS1d,BC3,BC1,CS1a,CS1b,CS1c
BC2,BC4,CS1d,BC3,BC1,CS1a,CS1c,CS1b
BC2,BC4,CS1d,BC3,BC1,CS1b,CS1a,CS1c
BC2,BC4,CS1d,BC3,BC1,CS1b,CS1c,CS1a
BC2,BC4,CS1d,BC3,BC1,CS1c,CS1a,CS1b
BC2,BC4,CS1d,BC3,BC1,CS1c,CS1b,CS1a
BC2,BC4,CS1d,BC3,CS1b,BC1,CS1a,CS1c
BC2,BC4,CS1d,BC3,CS1b,BC1,CS1c,CS1a
BC2,BC4,CS1d,BC3,CS1b,CS1c,BC1,CS1a
BC2,BC4,CS1d,BC3,CS1c,BC1,CS1a,CS1b
BC2,BC4,CS1d,BC3,CS1c,BC1,CS1b,CS1a
BC2,BC4,CS1d,BC3,CS1c,CS1b,BC1,CS1a
BC3,BC1,CS1a,BC2,CS1b,CS1c,BC4,CS1d
BC3,BC1,CS1a,BC2,CS1b,BC4,CS1c,CS1d
BC3,BC1,CS1a,BC2,CS1b,BC4,CS1d,CS1c
BC3,BC1,CS1a,BC2,CS1c,CS1b,BC4,CS1d
BC3,BC1,CS1a,BC2,CS1c,BC4,CS1b,CS1d
BC3,BC1,CS1a,BC2,CS1c,BC4,CS1d,CS1b
BC3,BC1,CS1a,BC2,BC4,CS1b,CS1c,CS1d
BC3,BC1,CS1a,BC2,BC4,CS1b,CS1d,CS1c
BC3,BC1,CS1a,BC2,BC4,CS1c,CS1b,CS1d
BC3,BC1,CS1a,BC2,BC4,CS1c,CS1d,CS1b
BC3,BC1,CS1a,BC2,BC4,CS1d,CS1b,CS1c
BC3,BC1,CS1a,BC2,BC4,CS1d,CS1c,CS1b
BC3,BC1,CS1a,CS1c,BC2,CS1b,BC4,CS1d
BC3,BC1,CS1a,CS1c,BC2,BC4,CS1b,CS1d
BC3,BC1,CS1a,CS1c,BC2,BC4,CS1d,CS1b
BC3,BC1,CS1a,CS1c,BC4,BC2,CS1b,CS1d
BC3,BC1,CS1a,CS1c,BC4,BC2,CS1d,CS1b
BC3,BC1,CS1a,CS1c,BC4,CS1d,BC2,CS1b
BC3,BC1,CS1a,BC4,BC2,CS1b,CS1c,CS1d
BC3,BC1,CS1a,BC4,BC2,CS1b,CS1d,CS1c
BC3,BC1,CS1a,BC4,BC2,CS1c,CS1b,CS1d
BC3,BC1,CS1a,BC4,BC2,CS1c,CS1d,CS1b
BC3,BC1,CS1a,BC4,BC2,CS1d,CS1b,CS1c
BC3,BC1,CS1a,BC4,BC2,CS1d,CS1c,CS1b

BC3,BC1,CS1a,BC4,CS1c,BC2,CS1b,CS1d
BC3,BC1,CS1a,BC4,CS1c,BC2,CS1d,CS1b
BC3,BC1,CS1a,BC4,CS1c,CS1d,BC2,CS1b
BC3,BC1,CS1a,BC4,CS1d,BC2,CS1b,CS1c
BC3,BC1,CS1a,BC4,CS1d,BC2,CS1c,CS1b
BC3,BC1,CS1a,BC4,CS1d,CS1c,BC2,CS1b
BC3,BC1,BC2,CS1a,CS1b,CS1c,BC4,CS1d
BC3,BC1,BC2,CS1a,CS1b,BC4,CS1c,CS1d
BC3,BC1,BC2,CS1a,CS1b,BC4,CS1d,CS1c
BC3,BC1,BC2,CS1a,CS1c,CS1b,BC4,CS1d
BC3,BC1,BC2,CS1a,CS1c,BC4,CS1b,CS1d
BC3,BC1,BC2,CS1a,CS1c,BC4,CS1d,CS1b
BC3,BC1,BC2,CS1a,BC4,CS1b,CS1c,CS1d
BC3,BC1,BC2,CS1a,BC4,CS1b,CS1d,CS1c
BC3,BC1,BC2,CS1a,BC4,CS1c,CS1b,CS1d
BC3,BC1,BC2,CS1a,BC4,CS1c,CS1d,CS1b
BC3,BC1,BC2,CS1a,BC4,CS1d,CS1b,CS1c
BC3,BC1,BC2,CS1a,BC4,CS1d,CS1c,CS1b
BC3,BC1,BC2,CS1b,CS1a,CS1c,BC4,CS1d
BC3,BC1,BC2,CS1b,CS1a,BC4,CS1c,CS1d
BC3,BC1,BC2,CS1b,CS1a,BC4,CS1d,CS1c
BC3,BC1,BC2,CS1b,CS1c,CS1a,BC4,CS1d
BC3,BC1,BC2,CS1b,CS1c,BC4,CS1a,CS1d
BC3,BC1,BC2,CS1b,CS1c,BC4,CS1d,CS1a
BC3,BC1,BC2,CS1b,BC4,CS1a,CS1c,CS1d
BC3,BC1,BC2,CS1b,BC4,CS1a,CS1d,CS1c
BC3,BC1,BC2,CS1b,BC4,CS1c,CS1a,CS1d
BC3,BC1,BC2,CS1b,BC4,CS1c,CS1d,CS1a
BC3,BC1,BC2,CS1b,BC4,CS1d,CS1a,CS1c
BC3,BC1,BC2,CS1b,BC4,CS1d,CS1c,CS1a
BC3,BC1,BC2,CS1c,CS1a,CS1b,BC4,CS1d
BC3,BC1,BC2,CS1c,CS1a,BC4,CS1b,CS1d
BC3,BC1,BC2,CS1c,CS1a,BC4,CS1d,CS1b
BC3,BC1,BC2,CS1c,CS1b,CS1a,BC4,CS1d
BC3,BC1,BC2,CS1c,CS1b,BC4,CS1a,CS1d
BC3,BC1,BC2,CS1c,CS1b,BC4,CS1d,CS1a
BC3,BC1,BC2,CS1c,BC4,CS1a,CS1b,CS1d
BC3,BC1,BC2,CS1c,BC4,CS1a,CS1d,CS1b
BC3,BC1,BC2,CS1c,BC4,CS1b,CS1a,CS1d
BC3,BC1,BC2,CS1c,BC4,CS1b,CS1d,CS1a
BC3,BC1,BC2,CS1c,BC4,CS1d,CS1a,CS1b
BC3,BC1,BC2,CS1c,BC4,CS1d,CS1b,CS1a
BC3,BC1,BC2,BC4,CS1a,CS1b,CS1c,CS1d
BC3,BC1,BC2,BC4,CS1a,CS1b,CS1d,CS1c
BC3,BC1,BC2,BC4,CS1a,CS1c,CS1b,CS1d
BC3,BC1,BC2,BC4,CS1a,CS1c,CS1d,CS1b

BC3,BC1,BC2,BC4,CS1a,CS1d,CS1b,CS1c
BC3,BC1,BC2,BC4,CS1a,CS1d,CS1c,CS1b
BC3,BC1,BC2,BC4,CS1b,CS1a,CS1c,CS1d
BC3,BC1,BC2,BC4,CS1b,CS1a,CS1d,CS1c
BC3,BC1,BC2,BC4,CS1b,CS1c,CS1a,CS1d
BC3,BC1,BC2,BC4,CS1b,CS1c,CS1d,CS1a
BC3,BC1,BC2,BC4,CS1b,CS1d,CS1a,CS1c
BC3,BC1,BC2,BC4,CS1b,CS1d,CS1c,CS1a
BC3,BC1,BC2,BC4,CS1c,CS1a,CS1b,CS1d
BC3,BC1,BC2,BC4,CS1c,CS1a,CS1d,CS1b
BC3,BC1,BC2,BC4,CS1c,CS1b,CS1a,CS1d
BC3,BC1,BC2,BC4,CS1c,CS1b,CS1d,CS1a
BC3,BC1,BC2,BC4,CS1c,CS1d,CS1a,CS1b
BC3,BC1,BC2,BC4,CS1c,CS1d,CS1b,CS1a
BC3,BC1,BC2,BC4,CS1d,CS1a,CS1b,CS1c
BC3,BC1,BC2,BC4,CS1d,CS1a,CS1c,CS1b
BC3,BC1,BC2,BC4,CS1d,CS1b,CS1a,CS1c
BC3,BC1,BC2,BC4,CS1d,CS1b,CS1c,CS1a
BC3,BC1,BC2,BC4,CS1d,CS1c,CS1a,CS1b
BC3,BC1,BC2,BC4,CS1d,CS1c,CS1b,CS1a
BC3,BC1,CS1c,CS1a,BC2,CS1b,BC4,CS1d
BC3,BC1,CS1c,CS1a,BC2,BC4,CS1b,CS1d
BC3,BC1,CS1c,CS1a,BC2,BC4,CS1d,CS1b
BC3,BC1,CS1c,CS1a,BC4,BC2,CS1b,CS1d
BC3,BC1,CS1c,CS1a,BC4,BC2,CS1d,CS1b
BC3,BC1,CS1c,CS1a,BC4,CS1d,BC2,CS1b
BC3,BC1,CS1c,BC2,CS1a,CS1b,BC4,CS1d
BC3,BC1,CS1c,BC2,CS1a,BC4,CS1b,CS1d
BC3,BC1,CS1c,BC2,CS1a,BC4,CS1d,CS1b
BC3,BC1,CS1c,BC2,CS1b,CS1a,BC4,CS1d
BC3,BC1,CS1c,BC2,CS1b,BC4,CS1a,CS1d
BC3,BC1,CS1c,BC2,CS1b,BC4,CS1d,CS1a
BC3,BC1,CS1c,BC2,BC4,CS1a,CS1b,CS1d
BC3,BC1,CS1c,BC2,BC4,CS1a,CS1d,CS1b
BC3,BC1,CS1c,BC2,BC4,CS1b,CS1a,CS1d
BC3,BC1,CS1c,BC2,BC4,CS1b,CS1d,CS1a
BC3,BC1,CS1c,BC2,BC4,CS1d,CS1a,CS1b
BC3,BC1,CS1c,BC2,BC4,CS1d,CS1b,CS1a
BC3,BC1,CS1c,BC4,CS1a,BC2,CS1b,CS1d
BC3,BC1,CS1c,BC4,CS1a,BC2,CS1d,CS1b
BC3,BC1,CS1c,BC4,CS1a,CS1d,BC2,CS1b
BC3,BC1,CS1c,BC4,BC2,CS1a,CS1b,CS1d
BC3,BC1,CS1c,BC4,BC2,CS1a,CS1d,CS1b
BC3,BC1,CS1c,BC4,BC2,CS1b,CS1a,CS1d
BC3,BC1,CS1c,BC4,BC2,CS1b,CS1d,CS1a
BC3,BC1,CS1c,BC4,BC2,CS1d,CS1a,CS1b

BC3,BC1,CS1c,BC4,BC2,CS1d,CS1b,CS1a
BC3,BC1,CS1c,BC4,CS1d,CS1a,BC2,CS1b
BC3,BC1,CS1c,BC4,CS1d,BC2,CS1a,CS1b
BC3,BC1,CS1c,BC4,CS1d,BC2,CS1b,CS1a
BC3,BC1,BC4,CS1a,BC2,CS1b,CS1c,CS1d
BC3,BC1,BC4,CS1a,BC2,CS1b,CS1d,CS1c
BC3,BC1,BC4,CS1a,BC2,CS1c,CS1b,CS1d
BC3,BC1,BC4,CS1a,BC2,CS1c,CS1d,CS1b
BC3,BC1,BC4,CS1a,BC2,CS1d,CS1b,CS1c
BC3,BC1,BC4,CS1a,BC2,CS1d,CS1c,CS1b
BC3,BC1,BC4,CS1a,CS1c,BC2,CS1b,CS1d
BC3,BC1,BC4,CS1a,CS1c,BC2,CS1d,CS1b
BC3,BC1,BC4,CS1a,CS1c,CS1d,BC2,CS1b
BC3,BC1,BC4,CS1a,CS1d,BC2,CS1b,CS1c
BC3,BC1,BC4,CS1a,CS1d,BC2,CS1c,CS1b
BC3,BC1,BC4,CS1a,CS1d,CS1c,BC2,CS1b
BC3,BC1,BC4,BC2,CS1a,CS1b,CS1c,CS1d
BC3,BC1,BC4,BC2,CS1a,CS1b,CS1d,CS1c
BC3,BC1,BC4,BC2,CS1a,CS1c,CS1b,CS1d
BC3,BC1,BC4,BC2,CS1a,CS1c,CS1d,CS1b
BC3,BC1,BC4,BC2,CS1a,CS1d,CS1b,CS1c
BC3,BC1,BC4,BC2,CS1a,CS1d,CS1c,CS1b
BC3,BC1,BC4,BC2,CS1b,CS1a,CS1c,CS1d
BC3,BC1,BC4,BC2,CS1b,CS1a,CS1d,CS1c
BC3,BC1,BC4,BC2,CS1b,CS1c,CS1a,CS1d
BC3,BC1,BC4,BC2,CS1b,CS1c,CS1d,CS1a
BC3,BC1,BC4,BC2,CS1b,CS1d,CS1a,CS1c
BC3,BC1,BC4,BC2,CS1b,CS1d,CS1c,CS1a
BC3,BC1,BC4,BC2,CS1c,CS1a,CS1b,CS1d
BC3,BC1,BC4,BC2,CS1c,CS1a,CS1d,CS1b
BC3,BC1,BC4,BC2,CS1c,CS1b,CS1a,CS1d
BC3,BC1,BC4,BC2,CS1c,CS1b,CS1d,CS1a
BC3,BC1,BC4,BC2,CS1c,CS1d,CS1a,CS1b
BC3,BC1,BC4,BC2,CS1c,CS1d,CS1b,CS1a
BC3,BC1,BC4,BC2,CS1d,CS1a,CS1b,CS1c
BC3,BC1,BC4,BC2,CS1d,CS1a,CS1c,CS1b
BC3,BC1,BC4,BC2,CS1d,CS1b,CS1a,CS1c
BC3,BC1,BC4,BC2,CS1d,CS1b,CS1c,CS1a
BC3,BC1,BC4,BC2,CS1d,CS1c,CS1a,CS1b
BC3,BC1,BC4,BC2,CS1d,CS1c,CS1b,CS1a
BC3,BC1,BC4,CS1c,CS1a,BC2,CS1b,CS1d
BC3,BC1,BC4,CS1c,CS1a,BC2,CS1d,CS1b
BC3,BC1,BC4,CS1c,CS1a,CS1d,BC2,CS1b
BC3,BC1,BC4,CS1c,BC2,CS1a,CS1b,CS1d
BC3,BC1,BC4,CS1c,BC2,CS1a,CS1d,CS1b
BC3,BC1,BC4,CS1c,BC2,CS1b,CS1a,CS1d

BC3,BC1,BC4,CS1c,BC2,CS1b,CS1d,CS1a
BC3,BC1,BC4,CS1c,BC2,CS1d,CS1a,CS1b
BC3,BC1,BC4,CS1c,BC2,CS1d,CS1b,CS1a
BC3,BC1,BC4,CS1c,CS1d,CS1a,BC2,CS1b
BC3,BC1,BC4,CS1c,CS1d,BC2,CS1a,CS1b
BC3,BC1,BC4,CS1c,CS1d,BC2,CS1b,CS1a
BC3,BC1,BC4,CS1d,CS1a,BC2,CS1b,CS1c
BC3,BC1,BC4,CS1d,CS1a,BC2,CS1c,CS1b
BC3,BC1,BC4,CS1d,CS1a,CS1c,BC2,CS1b
BC3,BC1,BC4,CS1d,BC2,CS1a,CS1b,CS1c
BC3,BC1,BC4,CS1d,BC2,CS1a,CS1c,CS1b
BC3,BC1,BC4,CS1d,BC2,CS1b,CS1a,CS1c
BC3,BC1,BC4,CS1d,BC2,CS1b,CS1c,CS1a
BC3,BC1,BC4,CS1d,BC2,CS1c,CS1a,CS1b
BC3,BC1,BC4,CS1d,BC2,CS1c,CS1b,CS1a
BC3,BC1,BC4,CS1d,CS1c,CS1a,BC2,CS1b
BC3,BC1,BC4,CS1d,CS1c,BC2,CS1a,CS1b
BC3,BC1,BC4,CS1d,CS1c,BC2,CS1b,CS1a
BC3,BC2,BC1,CS1a,CS1b,CS1c,BC4,CS1d
BC3,BC2,BC1,CS1a,CS1b,BC4,CS1c,CS1d
BC3,BC2,BC1,CS1a,CS1b,BC4,CS1d,CS1c
BC3,BC2,BC1,CS1a,CS1c,CS1b,BC4,CS1d
BC3,BC2,BC1,CS1a,CS1c,BC4,CS1b,CS1d
BC3,BC2,BC1,CS1a,CS1c,BC4,CS1d,CS1b
BC3,BC2,BC1,CS1a,BC4,CS1b,CS1c,CS1d
BC3,BC2,BC1,CS1a,BC4,CS1b,CS1d,CS1c
BC3,BC2,BC1,CS1a,BC4,CS1c,CS1b,CS1d
BC3,BC2,BC1,CS1a,BC4,CS1c,CS1d,CS1b
BC3,BC2,BC1,CS1a,BC4,CS1d,CS1b,CS1c
BC3,BC2,BC1,CS1a,BC4,CS1d,CS1c,CS1b
BC3,BC2,BC1,CS1b,CS1a,CS1c,BC4,CS1d
BC3,BC2,BC1,CS1b,CS1a,BC4,CS1c,CS1d
BC3,BC2,BC1,CS1b,CS1a,BC4,CS1d,CS1c
BC3,BC2,BC1,CS1b,CS1c,CS1a,BC4,CS1d
BC3,BC2,BC1,CS1b,CS1c,BC4,CS1a,CS1d
BC3,BC2,BC1,CS1b,CS1c,BC4,CS1d,CS1a
BC3,BC2,BC1,CS1b,BC4,CS1a,CS1c,CS1d
BC3,BC2,BC1,CS1b,BC4,CS1a,CS1d,CS1c
BC3,BC2,BC1,CS1b,BC4,CS1c,CS1a,CS1d
BC3,BC2,BC1,CS1b,BC4,CS1c,CS1d,CS1a
BC3,BC2,BC1,CS1b,BC4,CS1d,CS1a,CS1c
BC3,BC2,BC1,CS1b,BC4,CS1d,CS1c,CS1a
BC3,BC2,BC1,CS1c,CS1a,CS1b,BC4,CS1d
BC3,BC2,BC1,CS1c,CS1a,BC4,CS1b,CS1d
BC3,BC2,BC1,CS1c,CS1a,BC4,CS1d,CS1b
BC3,BC2,BC1,CS1c,CS1b,CS1a,BC4,CS1d

BC3,BC2,BC1,CS1c,CS1b,BC4,CS1a,CS1d
BC3,BC2,BC1,CS1c,CS1b,BC4,CS1d,CS1a
BC3,BC2,BC1,CS1c,BC4,CS1a,CS1b,CS1d
BC3,BC2,BC1,CS1c,BC4,CS1a,CS1d,CS1b
BC3,BC2,BC1,CS1c,BC4,CS1b,CS1a,CS1d
BC3,BC2,BC1,CS1c,BC4,CS1b,CS1d,CS1a
BC3,BC2,BC1,CS1c,BC4,CS1d,CS1a,CS1b
BC3,BC2,BC1,CS1c,BC4,CS1d,CS1b,CS1a
BC3,BC2,BC1,BC4,CS1a,CS1b,CS1c,CS1d
BC3,BC2,BC1,BC4,CS1a,CS1b,CS1d,CS1c
BC3,BC2,BC1,BC4,CS1a,CS1c,CS1b,CS1d
BC3,BC2,BC1,BC4,CS1a,CS1c,CS1d,CS1b
BC3,BC2,BC1,BC4,CS1a,CS1d,CS1b,CS1c
BC3,BC2,BC1,BC4,CS1a,CS1d,CS1c,CS1b
BC3,BC2,BC1,BC4,CS1b,CS1a,CS1c,CS1d
BC3,BC2,BC1,BC4,CS1b,CS1a,CS1d,CS1c
BC3,BC2,BC1,BC4,CS1b,CS1c,CS1a,CS1d
BC3,BC2,BC1,BC4,CS1b,CS1c,CS1d,CS1a
BC3,BC2,BC1,BC4,CS1b,CS1d,CS1a,CS1c
BC3,BC2,BC1,BC4,CS1b,CS1d,CS1c,CS1a
BC3,BC2,BC1,BC4,CS1c,CS1a,CS1b,CS1d
BC3,BC2,BC1,BC4,CS1c,CS1a,CS1d,CS1b
BC3,BC2,BC1,BC4,CS1c,CS1b,CS1a,CS1d
BC3,BC2,BC1,BC4,CS1c,CS1b,CS1d,CS1a
BC3,BC2,BC1,BC4,CS1c,CS1d,CS1a,CS1b
BC3,BC2,BC1,BC4,CS1c,CS1d,CS1b,CS1a
BC3,BC2,BC1,BC4,CS1d,CS1a,CS1b,CS1c
BC3,BC2,BC1,BC4,CS1d,CS1a,CS1c,CS1b
BC3,BC2,BC1,BC4,CS1d,CS1b,CS1a,CS1c
BC3,BC2,BC1,BC4,CS1d,CS1b,CS1c,CS1a
BC3,BC2,BC1,BC4,CS1d,CS1c,CS1a,CS1b
BC3,BC2,BC1,BC4,CS1d,CS1c,CS1b,CS1a
BC3,BC2,CS1b,BC1,CS1a,CS1c,BC4,CS1d
BC3,BC2,CS1b,BC1,CS1a,BC4,CS1c,CS1d
BC3,BC2,CS1b,BC1,CS1a,BC4,CS1d,CS1c
BC3,BC2,CS1b,BC1,CS1c,CS1a,BC4,CS1d
BC3,BC2,CS1b,BC1,CS1c,BC4,CS1a,CS1d
BC3,BC2,CS1b,BC1,CS1c,BC4,CS1d,CS1a
BC3,BC2,CS1b,BC1,BC4,CS1a,CS1c,CS1d
BC3,BC2,CS1b,BC1,BC4,CS1a,CS1d,CS1c
BC3,BC2,CS1b,BC1,BC4,CS1c,CS1a,CS1d
BC3,BC2,CS1b,BC1,BC4,CS1c,CS1d,CS1a
BC3,BC2,CS1b,BC1,BC4,CS1d,CS1a,CS1c
BC3,BC2,CS1b,BC1,BC4,CS1d,CS1c,CS1a
BC3,BC2,CS1b,CS1c,BC1,CS1a,BC4,CS1d
BC3,BC2,CS1b,CS1c,BC1,BC4,CS1a,CS1d

BC3,BC2,CS1b,CS1c,BC1,BC4,CS1d,CS1a
BC3,BC2,CS1b,CS1c,BC4,BC1,CS1a,CS1d
BC3,BC2,CS1b,CS1c,BC4,BC1,CS1d,CS1a
BC3,BC2,CS1b,CS1c,BC4,CS1d,BC1,CS1a
BC3,BC2,CS1b,BC4,BC1,CS1a,CS1c,CS1d
BC3,BC2,CS1b,BC4,BC1,CS1a,CS1d,CS1c
BC3,BC2,CS1b,BC4,BC1,CS1c,CS1a,CS1d
BC3,BC2,CS1b,BC4,BC1,CS1c,CS1d,CS1a
BC3,BC2,CS1b,BC4,BC1,CS1d,CS1a,CS1c
BC3,BC2,CS1b,BC4,BC1,CS1d,CS1c,CS1a
BC3,BC2,CS1b,BC4,CS1c,BC1,CS1a,CS1d
BC3,BC2,CS1b,BC4,CS1c,BC1,CS1d,CS1a
BC3,BC2,CS1b,BC4,CS1c,CS1d,BC1,CS1a
BC3,BC2,CS1b,BC4,CS1d,BC1,CS1a,CS1c
BC3,BC2,CS1b,BC4,CS1d,BC1,CS1c,CS1a
BC3,BC2,CS1b,BC4,CS1d,CS1c,BC1,CS1a
BC3,BC2,CS1c,BC1,CS1a,CS1b,BC4,CS1d
BC3,BC2,CS1c,BC1,CS1a,BC4,CS1b,CS1d
BC3,BC2,CS1c,BC1,CS1a,BC4,CS1d,CS1b
BC3,BC2,CS1c,BC1,CS1b,CS1a,BC4,CS1d
BC3,BC2,CS1c,BC1,CS1b,BC4,CS1a,CS1d
BC3,BC2,CS1c,BC1,CS1b,BC4,CS1d,CS1a
BC3,BC2,CS1c,BC1,BC4,CS1a,CS1b,CS1d
BC3,BC2,CS1c,BC1,BC4,CS1a,CS1d,CS1b
BC3,BC2,CS1c,BC1,BC4,CS1b,CS1a,CS1d
BC3,BC2,CS1c,BC1,BC4,CS1b,CS1d,CS1a
BC3,BC2,CS1c,BC1,BC4,CS1d,CS1a,CS1b
BC3,BC2,CS1c,BC1,BC4,CS1d,CS1b,CS1a
BC3,BC2,CS1c,CS1b,BC1,CS1a,BC4,CS1d
BC3,BC2,CS1c,CS1b,BC1,BC4,CS1a,CS1d
BC3,BC2,CS1c,CS1b,BC1,BC4,CS1d,CS1a
BC3,BC2,CS1c,CS1b,BC4,BC1,CS1a,CS1d
BC3,BC2,CS1c,CS1b,BC4,BC1,CS1d,CS1a
BC3,BC2,CS1c,CS1b,BC4,CS1d,BC1,CS1a
BC3,BC2,CS1c,BC4,BC1,CS1a,CS1b,CS1d
BC3,BC2,CS1c,BC4,BC1,CS1a,CS1d,CS1b
BC3,BC2,CS1c,BC4,BC1,CS1b,CS1a,CS1d
BC3,BC2,CS1c,BC4,BC1,CS1b,CS1d,CS1a
BC3,BC2,CS1c,BC4,BC1,CS1d,CS1a,CS1b
BC3,BC2,CS1c,BC4,BC1,CS1d,CS1b,CS1a
BC3,BC2,CS1c,BC4,CS1b,BC1,CS1a,CS1d
BC3,BC2,CS1c,BC4,CS1b,BC1,CS1d,CS1a
BC3,BC2,CS1c,BC4,CS1b,CS1d,BC1,CS1a
BC3,BC2,CS1c,BC4,CS1d,BC1,CS1a,CS1b
BC3,BC2,CS1c,BC4,CS1d,BC1,CS1b,CS1a
BC3,BC2,CS1c,BC4,CS1d,CS1b,BC1,CS1a

BC3,BC2,BC4,BC1,CS1a,CS1b,CS1c,CS1d
BC3,BC2,BC4,BC1,CS1a,CS1b,CS1d,CS1c
BC3,BC2,BC4,BC1,CS1a,CS1c,CS1b,CS1d
BC3,BC2,BC4,BC1,CS1a,CS1c,CS1d,CS1b
BC3,BC2,BC4,BC1,CS1a,CS1d,CS1b,CS1c
BC3,BC2,BC4,BC1,CS1a,CS1d,CS1c,CS1b
BC3,BC2,BC4,BC1,CS1b,CS1a,CS1c,CS1d
BC3,BC2,BC4,BC1,CS1b,CS1a,CS1d,CS1c
BC3,BC2,BC4,BC1,CS1b,CS1c,CS1a,CS1d
BC3,BC2,BC4,BC1,CS1b,CS1c,CS1d,CS1a
BC3,BC2,BC4,BC1,CS1b,CS1d,CS1a,CS1c
BC3,BC2,BC4,BC1,CS1b,CS1d,CS1c,CS1a
BC3,BC2,BC4,BC1,CS1c,CS1a,CS1b,CS1d
BC3,BC2,BC4,BC1,CS1c,CS1a,CS1d,CS1b
BC3,BC2,BC4,BC1,CS1c,CS1b,CS1a,CS1d
BC3,BC2,BC4,BC1,CS1c,CS1b,CS1d,CS1a
BC3,BC2,BC4,BC1,CS1c,CS1d,CS1a,CS1b
BC3,BC2,BC4,BC1,CS1c,CS1d,CS1b,CS1a
BC3,BC2,BC4,BC1,CS1d,CS1a,CS1b,CS1c
BC3,BC2,BC4,BC1,CS1d,CS1a,CS1c,CS1b
BC3,BC2,BC4,BC1,CS1d,CS1b,CS1a,CS1c
BC3,BC2,BC4,BC1,CS1d,CS1b,CS1c,CS1a
BC3,BC2,BC4,BC1,CS1d,CS1c,CS1a,CS1b
BC3,BC2,BC4,BC1,CS1d,CS1c,CS1b,CS1a
BC3,BC2,BC4,CS1b,BC1,CS1a,CS1c,CS1d
BC3,BC2,BC4,CS1b,BC1,CS1a,CS1d,CS1c
BC3,BC2,BC4,CS1b,BC1,CS1c,CS1a,CS1d
BC3,BC2,BC4,CS1b,BC1,CS1c,CS1d,CS1a
BC3,BC2,BC4,CS1b,BC1,CS1d,CS1a,CS1c
BC3,BC2,BC4,CS1b,BC1,CS1d,CS1c,CS1a
BC3,BC2,BC4,CS1b,CS1c,BC1,CS1a,CS1d
BC3,BC2,BC4,CS1b,CS1c,BC1,CS1d,CS1a
BC3,BC2,BC4,CS1b,CS1c,CS1d,BC1,CS1a
BC3,BC2,BC4,CS1b,CS1d,BC1,CS1a,CS1c
BC3,BC2,BC4,CS1b,CS1d,BC1,CS1c,CS1a
BC3,BC2,BC4,CS1b,CS1d,CS1c,BC1,CS1a
BC3,BC2,BC4,CS1c,BC1,CS1a,CS1b,CS1d
BC3,BC2,BC4,CS1c,BC1,CS1a,CS1d,CS1b
BC3,BC2,BC4,CS1c,BC1,CS1b,CS1a,CS1d
BC3,BC2,BC4,CS1c,BC1,CS1b,CS1d,CS1a
BC3,BC2,BC4,CS1c,BC1,CS1d,CS1a,CS1b
BC3,BC2,BC4,CS1c,BC1,CS1d,CS1b,CS1a
BC3,BC2,BC4,CS1c,CS1b,BC1,CS1a,CS1d
BC3,BC2,BC4,CS1c,CS1b,BC1,CS1d,CS1a
BC3,BC2,BC4,CS1c,CS1b,CS1d,BC1,CS1a
BC3,BC2,BC4,CS1c,CS1d,BC1,CS1a,CS1b

BC3,BC2,BC4,CS1c,CS1d,BC1,CS1b,CS1a
BC3,BC2,BC4,CS1c,CS1d,CS1b,BC1,CS1a
BC3,BC2,BC4,CS1d,BC1,CS1a,CS1b,CS1c
BC3,BC2,BC4,CS1d,BC1,CS1a,CS1c,CS1b
BC3,BC2,BC4,CS1d,BC1,CS1b,CS1a,CS1c
BC3,BC2,BC4,CS1d,BC1,CS1b,CS1c,CS1a
BC3,BC2,BC4,CS1d,BC1,CS1c,CS1a,CS1b
BC3,BC2,BC4,CS1d,BC1,CS1c,CS1b,CS1a
BC3,BC2,BC4,CS1d,CS1b,BC1,CS1a,CS1c
BC3,BC2,BC4,CS1d,CS1b,BC1,CS1c,CS1a
BC3,BC2,BC4,CS1d,CS1b,CS1c,BC1,CS1a
BC3,BC2,BC4,CS1d,CS1c,BC1,CS1a,CS1b
BC3,BC2,BC4,CS1d,CS1c,BC1,CS1b,CS1a
BC3,BC2,BC4,CS1d,CS1c,CS1b,BC1,CS1a
BC3,CS1c,BC1,CS1a,BC2,CS1b,BC4,CS1d
BC3,CS1c,BC1,CS1a,BC2,BC4,CS1b,CS1d
BC3,CS1c,BC1,CS1a,BC2,BC4,CS1d,CS1b
BC3,CS1c,BC1,CS1a,BC4,BC2,CS1b,CS1d
BC3,CS1c,BC1,CS1a,BC4,BC2,CS1d,CS1b
BC3,CS1c,BC1,CS1a,BC4,CS1d,BC2,CS1b
BC3,CS1c,BC1,BC2,CS1a,CS1b,BC4,CS1d
BC3,CS1c,BC1,BC2,CS1a,BC4,CS1b,CS1d
BC3,CS1c,BC1,BC2,CS1a,BC4,CS1d,CS1b
BC3,CS1c,BC1,BC2,CS1b,CS1a,BC4,CS1d
BC3,CS1c,BC1,BC2,CS1b,BC4,CS1a,CS1d
BC3,CS1c,BC1,BC2,CS1b,BC4,CS1d,CS1a
BC3,CS1c,BC1,BC2,BC4,CS1a,CS1b,CS1d
BC3,CS1c,BC1,BC2,BC4,CS1a,CS1d,CS1b
BC3,CS1c,BC1,BC2,BC4,CS1b,CS1a,CS1d
BC3,CS1c,BC1,BC2,BC4,CS1b,CS1d,CS1a
BC3,CS1c,BC1,BC2,BC4,CS1d,CS1a,CS1b
BC3,CS1c,BC1,BC2,BC4,CS1d,CS1b,CS1a
BC3,CS1c,BC1,BC4,CS1a,BC2,CS1b,CS1d
BC3,CS1c,BC1,BC4,CS1a,BC2,CS1d,CS1b
BC3,CS1c,BC1,BC4,CS1a,CS1d,BC2,CS1b
BC3,CS1c,BC1,BC4,BC2,CS1a,CS1b,CS1d
BC3,CS1c,BC1,BC4,BC2,CS1a,CS1d,CS1b
BC3,CS1c,BC1,BC4,BC2,CS1b,CS1a,CS1d
BC3,CS1c,BC1,BC4,BC2,CS1b,CS1d,CS1a
BC3,CS1c,BC1,BC4,BC2,CS1d,CS1a,CS1b
BC3,CS1c,BC1,BC4,BC2,CS1d,CS1b,CS1a
BC3,CS1c,BC1,BC4,CS1d,CS1a,BC2,CS1b
BC3,CS1c,BC1,BC4,CS1d,BC2,CS1a,CS1b
BC3,CS1c,BC1,BC4,CS1d,BC2,CS1b,CS1a
BC3,CS1c,BC2,BC1,CS1a,CS1b,BC4,CS1d
BC3,CS1c,BC2,BC1,CS1a,BC4,CS1b,CS1d

BC3,CS1c,BC2,BC1,CS1a,BC4,CS1d,CS1b
BC3,CS1c,BC2,BC1,CS1b,CS1a,BC4,CS1d
BC3,CS1c,BC2,BC1,CS1b,BC4,CS1a,CS1d
BC3,CS1c,BC2,BC1,CS1b,BC4,CS1d,CS1a
BC3,CS1c,BC2,BC1,BC4,CS1a,CS1b,CS1d
BC3,CS1c,BC2,BC1,BC4,CS1a,CS1d,CS1b
BC3,CS1c,BC2,BC1,BC4,CS1b,CS1a,CS1d
BC3,CS1c,BC2,BC1,BC4,CS1b,CS1d,CS1a
BC3,CS1c,BC2,BC1,BC4,CS1d,CS1a,CS1b
BC3,CS1c,BC2,BC1,BC4,CS1d,CS1b,CS1a
BC3,CS1c,BC2,CS1b,BC1,CS1a,BC4,CS1d
BC3,CS1c,BC2,CS1b,BC1,BC4,CS1a,CS1d
BC3,CS1c,BC2,CS1b,BC1,BC4,CS1d,CS1a
BC3,CS1c,BC2,CS1b,BC4,BC1,CS1a,CS1d
BC3,CS1c,BC2,CS1b,BC4,BC1,CS1d,CS1a
BC3,CS1c,BC2,CS1b,BC4,CS1d,BC1,CS1a
BC3,CS1c,BC2,BC4,BC1,CS1a,CS1b,CS1d
BC3,CS1c,BC2,BC4,BC1,CS1a,CS1d,CS1b
BC3,CS1c,BC2,BC4,BC1,CS1b,CS1a,CS1d
BC3,CS1c,BC2,BC4,BC1,CS1b,CS1d,CS1a
BC3,CS1c,BC2,BC4,BC1,CS1d,CS1a,CS1b
BC3,CS1c,BC2,BC4,BC1,CS1d,CS1b,CS1a
BC3,CS1c,BC2,BC4,CS1b,BC1,CS1a,CS1d
BC3,CS1c,BC2,BC4,CS1b,BC1,CS1d,CS1a
BC3,CS1c,BC2,BC4,CS1b,CS1d,BC1,CS1a
BC3,CS1c,BC2,BC4,CS1d,BC1,CS1a,CS1b
BC3,CS1c,BC2,BC4,CS1d,BC1,CS1b,CS1a
BC3,CS1c,BC2,BC4,CS1d,CS1b,BC1,CS1a
BC3,CS1c,BC4,BC1,CS1a,BC2,CS1b,CS1d
BC3,CS1c,BC4,BC1,CS1a,BC2,CS1d,CS1b
BC3,CS1c,BC4,BC1,CS1a,CS1d,BC2,CS1b
BC3,CS1c,BC4,BC1,BC2,CS1a,CS1b,CS1d
BC3,CS1c,BC4,BC1,BC2,CS1a,CS1d,CS1b
BC3,CS1c,BC4,BC1,BC2,CS1b,CS1a,CS1d
BC3,CS1c,BC4,BC1,BC2,CS1b,CS1d,CS1a
BC3,CS1c,BC4,BC1,BC2,CS1d,CS1a,CS1b
BC3,CS1c,BC4,BC1,BC2,CS1d,CS1b,CS1a
BC3,CS1c,BC4,BC1,CS1d,CS1a,BC2,CS1b
BC3,CS1c,BC4,BC1,CS1d,BC2,CS1a,CS1b
BC3,CS1c,BC4,BC1,CS1d,BC2,CS1b,CS1a
BC3,CS1c,BC4,BC2,BC1,CS1a,CS1b,CS1d
BC3,CS1c,BC4,BC2,BC1,CS1a,CS1d,CS1b
BC3,CS1c,BC4,BC2,BC1,CS1b,CS1a,CS1d
BC3,CS1c,BC4,BC2,BC1,CS1b,CS1d,CS1a
BC3,CS1c,BC4,BC2,BC1,CS1d,CS1a,CS1b
BC3,CS1c,BC4,BC2,BC1,CS1d,CS1b,CS1a

BC3,CS1c,BC4,BC2,CS1b,BC1,CS1a,CS1d
BC3,CS1c,BC4,BC2,CS1b,BC1,CS1d,CS1a
BC3,CS1c,BC4,BC2,CS1b,CS1d,BC1,CS1a
BC3,CS1c,BC4,BC2,CS1d,BC1,CS1a,CS1b
BC3,CS1c,BC4,BC2,CS1d,BC1,CS1b,CS1a
BC3,CS1c,BC4,BC2,CS1d,CS1b,BC1,CS1a
BC3,CS1c,BC4,CS1d,BC1,CS1a,BC2,CS1b
BC3,CS1c,BC4,CS1d,BC1,BC2,CS1a,CS1b
BC3,CS1c,BC4,CS1d,BC1,BC2,CS1b,CS1a
BC3,CS1c,BC4,CS1d,BC2,BC1,CS1a,CS1b
BC3,CS1c,BC4,CS1d,BC2,BC1,CS1b,CS1a
BC3,CS1c,BC4,CS1d,BC2,CS1b,BC1,CS1a
BC3,BC4,BC1,CS1a,BC2,CS1b,CS1c,CS1d
BC3,BC4,BC1,CS1a,BC2,CS1b,CS1d,CS1c
BC3,BC4,BC1,CS1a,BC2,CS1c,CS1b,CS1d
BC3,BC4,BC1,CS1a,BC2,CS1c,CS1d,CS1b
BC3,BC4,BC1,CS1a,BC2,CS1d,CS1b,CS1c
BC3,BC4,BC1,CS1a,BC2,CS1d,CS1c,CS1b
BC3,BC4,BC1,CS1a,CS1c,BC2,CS1b,CS1d
BC3,BC4,BC1,CS1a,CS1c,BC2,CS1d,CS1b
BC3,BC4,BC1,CS1a,CS1c,CS1d,BC2,CS1b
BC3,BC4,BC1,CS1a,CS1d,BC2,CS1b,CS1c
BC3,BC4,BC1,CS1a,CS1d,BC2,CS1c,CS1b
BC3,BC4,BC1,CS1a,CS1d,CS1c,BC2,CS1b
BC3,BC4,BC1,BC2,CS1a,CS1b,CS1c,CS1d
BC3,BC4,BC1,BC2,CS1a,CS1b,CS1d,CS1c
BC3,BC4,BC1,BC2,CS1a,CS1c,CS1b,CS1d
BC3,BC4,BC1,BC2,CS1a,CS1c,CS1d,CS1b
BC3,BC4,BC1,BC2,CS1a,CS1d,CS1b,CS1c
BC3,BC4,BC1,BC2,CS1a,CS1d,CS1c,CS1b
BC3,BC4,BC1,BC2,CS1b,CS1a,CS1c,CS1d
BC3,BC4,BC1,BC2,CS1b,CS1a,CS1d,CS1c
BC3,BC4,BC1,BC2,CS1b,CS1c,CS1a,CS1d
BC3,BC4,BC1,BC2,CS1b,CS1c,CS1d,CS1a
BC3,BC4,BC1,BC2,CS1b,CS1d,CS1a,CS1c
BC3,BC4,BC1,BC2,CS1b,CS1d,CS1c,CS1a
BC3,BC4,BC1,BC2,CS1c,CS1a,CS1b,CS1d
BC3,BC4,BC1,BC2,CS1c,CS1a,CS1d,CS1b
BC3,BC4,BC1,BC2,CS1c,CS1b,CS1a,CS1d
BC3,BC4,BC1,BC2,CS1c,CS1b,CS1d,CS1a
BC3,BC4,BC1,BC2,CS1c,CS1d,CS1a,CS1b
BC3,BC4,BC1,BC2,CS1c,CS1d,CS1b,CS1a
BC3,BC4,BC1,BC2,CS1d,CS1a,CS1b,CS1c
BC3,BC4,BC1,BC2,CS1d,CS1a,CS1c,CS1b
BC3,BC4,BC1,BC2,CS1d,CS1b,CS1a,CS1c
BC3,BC4,BC1,BC2,CS1d,CS1b,CS1c,CS1a

BC3,BC4,BC1,BC2,CS1d,CS1c,CS1a,CS1b
BC3,BC4,BC1,BC2,CS1d,CS1c,CS1b,CS1a
BC3,BC4,BC1,CS1c,CS1a,BC2,CS1b,CS1d
BC3,BC4,BC1,CS1c,CS1a,BC2,CS1d,CS1b
BC3,BC4,BC1,CS1c,CS1a,CS1d,BC2,CS1b
BC3,BC4,BC1,CS1c,BC2,CS1a,CS1b,CS1d
BC3,BC4,BC1,CS1c,BC2,CS1a,CS1d,CS1b
BC3,BC4,BC1,CS1c,BC2,CS1b,CS1a,CS1d
BC3,BC4,BC1,CS1c,BC2,CS1b,CS1d,CS1a
BC3,BC4,BC1,CS1c,BC2,CS1d,CS1a,CS1b
BC3,BC4,BC1,CS1c,BC2,CS1d,CS1b,CS1a
BC3,BC4,BC1,CS1c,CS1d,CS1a,BC2,CS1b
BC3,BC4,BC1,CS1c,CS1d,BC2,CS1a,CS1b
BC3,BC4,BC1,CS1c,CS1d,BC2,CS1b,CS1a
BC3,BC4,BC1,CS1d,CS1a,BC2,CS1b,CS1c
BC3,BC4,BC1,CS1d,CS1a,BC2,CS1c,CS1b
BC3,BC4,BC1,CS1d,CS1a,CS1c,BC2,CS1b
BC3,BC4,BC1,CS1d,BC2,CS1a,CS1b,CS1c
BC3,BC4,BC1,CS1d,BC2,CS1a,CS1c,CS1b
BC3,BC4,BC1,CS1d,BC2,CS1b,CS1a,CS1c
BC3,BC4,BC1,CS1d,BC2,CS1b,CS1c,CS1a
BC3,BC4,BC1,CS1d,BC2,CS1c,CS1a,CS1b
BC3,BC4,BC1,CS1d,BC2,CS1c,CS1b,CS1a
BC3,BC4,BC1,CS1d,CS1c,CS1a,BC2,CS1b
BC3,BC4,BC1,CS1d,CS1c,BC2,CS1a,CS1b
BC3,BC4,BC1,CS1d,CS1c,BC2,CS1b,CS1a
BC3,BC4,BC2,BC1,CS1a,CS1b,CS1c,CS1d
BC3,BC4,BC2,BC1,CS1a,CS1b,CS1d,CS1c
BC3,BC4,BC2,BC1,CS1a,CS1c,CS1b,CS1d
BC3,BC4,BC2,BC1,CS1a,CS1c,CS1d,CS1b
BC3,BC4,BC2,BC1,CS1a,CS1d,CS1b,CS1c
BC3,BC4,BC2,BC1,CS1a,CS1d,CS1c,CS1b
BC3,BC4,BC2,BC1,CS1b,CS1a,CS1c,CS1d
BC3,BC4,BC2,BC1,CS1b,CS1a,CS1d,CS1c
BC3,BC4,BC2,BC1,CS1b,CS1c,CS1a,CS1d
BC3,BC4,BC2,BC1,CS1b,CS1c,CS1d,CS1a
BC3,BC4,BC2,BC1,CS1b,CS1d,CS1a,CS1c
BC3,BC4,BC2,BC1,CS1b,CS1d,CS1c,CS1a
BC3,BC4,BC2,BC1,CS1c,CS1a,CS1b,CS1d
BC3,BC4,BC2,BC1,CS1c,CS1a,CS1d,CS1b
BC3,BC4,BC2,BC1,CS1c,CS1b,CS1a,CS1d
BC3,BC4,BC2,BC1,CS1c,CS1b,CS1d,CS1a
BC3,BC4,BC2,BC1,CS1c,CS1d,CS1a,CS1b
BC3,BC4,BC2,BC1,CS1c,CS1d,CS1b,CS1a
BC3,BC4,BC2,BC1,CS1d,CS1a,CS1b,CS1c
BC3,BC4,BC2,BC1,CS1d,CS1a,CS1c,CS1b

BC3,BC4,BC2,BC1,CS1d,CS1b,CS1a,CS1c
BC3,BC4,BC2,BC1,CS1d,CS1b,CS1c,CS1a
BC3,BC4,BC2,BC1,CS1d,CS1c,CS1a,CS1b
BC3,BC4,BC2,BC1,CS1d,CS1c,CS1b,CS1a
BC3,BC4,BC2,CS1b,BC1,CS1a,CS1c,CS1d
BC3,BC4,BC2,CS1b,BC1,CS1a,CS1d,CS1c
BC3,BC4,BC2,CS1b,BC1,CS1c,CS1a,CS1d
BC3,BC4,BC2,CS1b,BC1,CS1c,CS1d,CS1a
BC3,BC4,BC2,CS1b,BC1,CS1d,CS1a,CS1c
BC3,BC4,BC2,CS1b,BC1,CS1d,CS1c,CS1a
BC3,BC4,BC2,CS1b,CS1c,BC1,CS1a,CS1d
BC3,BC4,BC2,CS1b,CS1c,BC1,CS1d,CS1a
BC3,BC4,BC2,CS1b,CS1c,CS1d,BC1,CS1a
BC3,BC4,BC2,CS1b,CS1d,BC1,CS1a,CS1c
BC3,BC4,BC2,CS1b,CS1d,BC1,CS1c,CS1a
BC3,BC4,BC2,CS1b,CS1d,CS1c,BC1,CS1a
BC3,BC4,BC2,CS1c,BC1,CS1a,CS1b,CS1d
BC3,BC4,BC2,CS1c,BC1,CS1a,CS1d,CS1b
BC3,BC4,BC2,CS1c,BC1,CS1b,CS1a,CS1d
BC3,BC4,BC2,CS1c,BC1,CS1b,CS1d,CS1a
BC3,BC4,BC2,CS1c,BC1,CS1d,CS1a,CS1b
BC3,BC4,BC2,CS1c,BC1,CS1d,CS1b,CS1a
BC3,BC4,BC2,CS1c,CS1b,BC1,CS1a,CS1d
BC3,BC4,BC2,CS1c,CS1b,BC1,CS1d,CS1a
BC3,BC4,BC2,CS1c,CS1b,CS1d,BC1,CS1a
BC3,BC4,BC2,CS1c,CS1d,BC1,CS1a,CS1b
BC3,BC4,BC2,CS1c,CS1d,BC1,CS1b,CS1a
BC3,BC4,BC2,CS1c,CS1d,CS1b,BC1,CS1a
BC3,BC4,BC2,CS1d,BC1,CS1a,CS1b,CS1c
BC3,BC4,BC2,CS1d,BC1,CS1a,CS1c,CS1b
BC3,BC4,BC2,CS1d,BC1,CS1b,CS1a,CS1c
BC3,BC4,BC2,CS1d,BC1,CS1b,CS1c,CS1a
BC3,BC4,BC2,CS1d,BC1,CS1c,CS1a,CS1b
BC3,BC4,BC2,CS1d,BC1,CS1c,CS1b,CS1a
BC3,BC4,BC2,CS1d,CS1b,BC1,CS1a,CS1c
BC3,BC4,BC2,CS1d,CS1b,BC1,CS1c,CS1a
BC3,BC4,BC2,CS1d,CS1b,CS1c,BC1,CS1a
BC3,BC4,BC2,CS1d,CS1c,BC1,CS1a,CS1b
BC3,BC4,BC2,CS1d,CS1c,BC1,CS1b,CS1a
BC3,BC4,BC2,CS1d,CS1c,CS1b,BC1,CS1a
BC3,BC4,CS1c,BC1,CS1a,BC2,CS1b,CS1d
BC3,BC4,CS1c,BC1,CS1a,BC2,CS1d,CS1b
BC3,BC4,CS1c,BC1,CS1a,CS1d,BC2,CS1b
BC3,BC4,CS1c,BC1,BC2,CS1a,CS1b,CS1d
BC3,BC4,CS1c,BC1,BC2,CS1a,CS1d,CS1b
BC3,BC4,CS1c,BC1,BC2,CS1b,CS1a,CS1d

BC3,BC4,CS1c,BC1,BC2,CS1b,CS1d,CS1a
BC3,BC4,CS1c,BC1,BC2,CS1d,CS1a,CS1b
BC3,BC4,CS1c,BC1,BC2,CS1d,CS1b,CS1a
BC3,BC4,CS1c,BC1,CS1d,CS1a,BC2,CS1b
BC3,BC4,CS1c,BC1,CS1d,BC2,CS1a,CS1b
BC3,BC4,CS1c,BC1,CS1d,BC2,CS1b,CS1a
BC3,BC4,CS1c,BC2,BC1,CS1a,CS1b,CS1d
BC3,BC4,CS1c,BC2,BC1,CS1a,CS1d,CS1b
BC3,BC4,CS1c,BC2,BC1,CS1b,CS1a,CS1d
BC3,BC4,CS1c,BC2,BC1,CS1b,CS1d,CS1a
BC3,BC4,CS1c,BC2,BC1,CS1d,CS1a,CS1b
BC3,BC4,CS1c,BC2,BC1,CS1d,CS1b,CS1a
BC3,BC4,CS1c,BC2,CS1b,BC1,CS1a,CS1d
BC3,BC4,CS1c,BC2,CS1b,BC1,CS1d,CS1a
BC3,BC4,CS1c,BC2,CS1b,CS1d,BC1,CS1a
BC3,BC4,CS1c,BC2,CS1d,BC1,CS1a,CS1b
BC3,BC4,CS1c,BC2,CS1d,BC1,CS1b,CS1a
BC3,BC4,CS1c,BC2,CS1d,CS1b,BC1,CS1a
BC3,BC4,CS1c,CS1d,BC1,CS1a,BC2,CS1b
BC3,BC4,CS1c,CS1d,BC1,BC2,CS1a,CS1b
BC3,BC4,CS1c,CS1d,BC1,BC2,CS1b,CS1a
BC3,BC4,CS1c,CS1d,BC2,BC1,CS1a,CS1b
BC3,BC4,CS1c,CS1d,BC2,BC1,CS1b,CS1a
BC3,BC4,CS1c,CS1d,BC2,CS1b,BC1,CS1a
BC3,BC4,CS1d,BC1,CS1a,BC2,CS1b,CS1c
BC3,BC4,CS1d,BC1,CS1a,BC2,CS1c,CS1b
BC3,BC4,CS1d,BC1,CS1a,CS1c,BC2,CS1b
BC3,BC4,CS1d,BC1,BC2,CS1a,CS1b,CS1c
BC3,BC4,CS1d,BC1,BC2,CS1a,CS1c,CS1b
BC3,BC4,CS1d,BC1,BC2,CS1b,CS1a,CS1c
BC3,BC4,CS1d,BC1,BC2,CS1b,CS1c,CS1a
BC3,BC4,CS1d,BC1,BC2,CS1c,CS1a,CS1b
BC3,BC4,CS1d,BC1,BC2,CS1c,CS1b,CS1a
BC3,BC4,CS1d,BC1,CS1c,CS1a,BC2,CS1b
BC3,BC4,CS1d,BC1,CS1c,BC2,CS1a,CS1b
BC3,BC4,CS1d,BC1,CS1c,BC2,CS1b,CS1a
BC3,BC4,CS1d,BC2,BC1,CS1a,CS1b,CS1c
BC3,BC4,CS1d,BC2,BC1,CS1a,CS1c,CS1b
BC3,BC4,CS1d,BC2,BC1,CS1b,CS1a,CS1c
BC3,BC4,CS1d,BC2,BC1,CS1b,CS1c,CS1a
BC3,BC4,CS1d,BC2,BC1,CS1c,CS1a,CS1b
BC3,BC4,CS1d,BC2,BC1,CS1c,CS1b,CS1a
BC3,BC4,CS1d,BC2,CS1b,BC1,CS1a,CS1c
BC3,BC4,CS1d,BC2,CS1b,BC1,CS1c,CS1a
BC3,BC4,CS1d,BC2,CS1b,CS1c,BC1,CS1a
BC3,BC4,CS1d,BC2,CS1c,BC1,CS1a,CS1b

BC3,BC4,CS1d,BC2,CS1c,BC1,CS1b,CS1a
BC3,BC4,CS1d,BC2,CS1c,CS1b,BC1,CS1a
BC3,BC4,CS1d,CS1c,BC1,CS1a,BC2,CS1b
BC3,BC4,CS1d,CS1c,BC1,BC2,CS1a,CS1b
BC3,BC4,CS1d,CS1c,BC1,BC2,CS1b,CS1a
BC3,BC4,CS1d,CS1c,BC2,BC1,CS1a,CS1b
BC3,BC4,CS1d,CS1c,BC2,BC1,CS1b,CS1a
BC3,BC4,CS1d,CS1c,BC2,CS1b,BC1,CS1a
BC4,BC1,CS1a,BC2,CS1b,BC3,CS1c,CS1d
BC4,BC1,CS1a,BC2,CS1b,BC3,CS1d,CS1c
BC4,BC1,CS1a,BC2,CS1b,CS1d,BC3,CS1c
BC4,BC1,CS1a,BC2,BC3,CS1b,CS1c,CS1d
BC4,BC1,CS1a,BC2,BC3,CS1b,CS1d,CS1c
BC4,BC1,CS1a,BC2,BC3,CS1c,CS1b,CS1d
BC4,BC1,CS1a,BC2,BC3,CS1c,CS1d,CS1b
BC4,BC1,CS1a,BC2,BC3,CS1d,CS1b,CS1c
BC4,BC1,CS1a,BC2,BC3,CS1d,CS1c,CS1b
BC4,BC1,CS1a,BC2,CS1d,CS1b,BC3,CS1c
BC4,BC1,CS1a,BC2,CS1d,BC3,CS1b,CS1c
BC4,BC1,CS1a,BC2,CS1d,BC3,CS1c,CS1b
BC4,BC1,CS1a,BC3,BC2,CS1b,CS1c,CS1d
BC4,BC1,CS1a,BC3,BC2,CS1b,CS1d,CS1c
BC4,BC1,CS1a,BC3,BC2,CS1c,CS1b,CS1d
BC4,BC1,CS1a,BC3,BC2,CS1c,CS1d,CS1b
BC4,BC1,CS1a,BC3,BC2,CS1d,CS1b,CS1c
BC4,BC1,CS1a,BC3,BC2,CS1d,CS1c,CS1b
BC4,BC1,CS1a,BC3,CS1c,BC2,CS1b,CS1d
BC4,BC1,CS1a,BC3,CS1c,BC2,CS1d,CS1b
BC4,BC1,CS1a,BC3,CS1c,CS1d,BC2,CS1b
BC4,BC1,CS1a,BC3,CS1d,BC2,CS1b,CS1c
BC4,BC1,CS1a,BC3,CS1d,BC2,CS1c,CS1b
BC4,BC1,CS1a,BC3,CS1d,CS1c,BC2,CS1b
BC4,BC1,CS1a,CS1d,BC2,CS1b,BC3,CS1c
BC4,BC1,CS1a,CS1d,BC2,BC3,CS1b,CS1c
BC4,BC1,CS1a,CS1d,BC2,BC3,CS1c,CS1b
BC4,BC1,CS1a,CS1d,BC3,BC2,CS1b,CS1c
BC4,BC1,CS1a,CS1d,BC3,BC2,CS1c,CS1b
BC4,BC1,CS1a,CS1d,BC3,CS1c,BC2,CS1b
BC4,BC1,BC2,CS1a,CS1b,BC3,CS1c,CS1d
BC4,BC1,BC2,CS1a,CS1b,BC3,CS1d,CS1c
BC4,BC1,BC2,CS1a,CS1b,CS1d,BC3,CS1c
BC4,BC1,BC2,CS1a,BC3,CS1b,CS1c,CS1d
BC4,BC1,BC2,CS1a,BC3,CS1b,CS1d,CS1c
BC4,BC1,BC2,CS1a,BC3,CS1c,CS1b,CS1d
BC4,BC1,BC2,CS1a,BC3,CS1c,CS1d,CS1b
BC4,BC1,BC2,CS1a,BC3,CS1d,CS1b,CS1c

BC4,BC1,BC2,CS1a,BC3,CS1d,CS1c,CS1b
BC4,BC1,BC2,CS1a,CS1d,CS1b,BC3,CS1c
BC4,BC1,BC2,CS1a,CS1d,BC3,CS1b,CS1c
BC4,BC1,BC2,CS1a,CS1d,BC3,CS1c,CS1b
BC4,BC1,BC2,CS1b,CS1a,BC3,CS1c,CS1d
BC4,BC1,BC2,CS1b,CS1a,BC3,CS1d,CS1c
BC4,BC1,BC2,CS1b,CS1a,CS1d,BC3,CS1c
BC4,BC1,BC2,CS1b,BC3,CS1a,CS1c,CS1d
BC4,BC1,BC2,CS1b,BC3,CS1a,CS1d,CS1c
BC4,BC1,BC2,CS1b,BC3,CS1c,CS1a,CS1d
BC4,BC1,BC2,CS1b,BC3,CS1c,CS1d,CS1a
BC4,BC1,BC2,CS1b,BC3,CS1d,CS1a,CS1c
BC4,BC1,BC2,CS1b,BC3,CS1d,CS1c,CS1a
BC4,BC1,BC2,CS1b,CS1d,CS1a,BC3,CS1c
BC4,BC1,BC2,CS1b,CS1d,BC3,CS1a,CS1c
BC4,BC1,BC2,CS1b,CS1d,BC3,CS1c,CS1a
BC4,BC1,BC2,BC3,CS1a,CS1b,CS1c,CS1d
BC4,BC1,BC2,BC3,CS1a,CS1b,CS1d,CS1c
BC4,BC1,BC2,BC3,CS1a,CS1c,CS1b,CS1d
BC4,BC1,BC2,BC3,CS1a,CS1c,CS1d,CS1b
BC4,BC1,BC2,BC3,CS1a,CS1d,CS1b,CS1c
BC4,BC1,BC2,BC3,CS1a,CS1d,CS1c,CS1b
BC4,BC1,BC2,BC3,CS1b,CS1a,CS1c,CS1d
BC4,BC1,BC2,BC3,CS1b,CS1a,CS1d,CS1c
BC4,BC1,BC2,BC3,CS1b,CS1c,CS1a,CS1d
BC4,BC1,BC2,BC3,CS1b,CS1c,CS1d,CS1a
BC4,BC1,BC2,BC3,CS1b,CS1d,CS1a,CS1c
BC4,BC1,BC2,BC3,CS1b,CS1d,CS1c,CS1a
BC4,BC1,BC2,BC3,CS1c,CS1a,CS1b,CS1d
BC4,BC1,BC2,BC3,CS1c,CS1a,CS1d,CS1b
BC4,BC1,BC2,BC3,CS1c,CS1b,CS1a,CS1d
BC4,BC1,BC2,BC3,CS1c,CS1b,CS1d,CS1a
BC4,BC1,BC2,BC3,CS1c,CS1d,CS1a,CS1b
BC4,BC1,BC2,BC3,CS1c,CS1d,CS1b,CS1a
BC4,BC1,BC2,BC3,CS1d,CS1a,CS1b,CS1c
BC4,BC1,BC2,BC3,CS1d,CS1a,CS1c,CS1b
BC4,BC1,BC2,BC3,CS1d,CS1b,CS1a,CS1c
BC4,BC1,BC2,BC3,CS1d,CS1b,CS1c,CS1a
BC4,BC1,BC2,BC3,CS1d,CS1c,CS1a,CS1b
BC4,BC1,BC2,BC3,CS1d,CS1c,CS1b,CS1a
BC4,BC1,BC2,CS1d,CS1a,CS1b,BC3,CS1c
BC4,BC1,BC2,CS1d,CS1a,BC3,CS1b,CS1c
BC4,BC1,BC2,CS1d,CS1a,BC3,CS1c,CS1b
BC4,BC1,BC2,CS1d,CS1b,CS1a,BC3,CS1c
BC4,BC1,BC2,CS1d,CS1b,BC3,CS1a,CS1c
BC4,BC1,BC2,CS1d,CS1b,BC3,CS1c,CS1a

BC4,BC1,BC2,CS1d,BC3,CS1a,CS1b,CS1c
BC4,BC1,BC2,CS1d,BC3,CS1a,CS1c,CS1b
BC4,BC1,BC2,CS1d,BC3,CS1b,CS1a,CS1c
BC4,BC1,BC2,CS1d,BC3,CS1b,CS1c,CS1a
BC4,BC1,BC2,CS1d,BC3,CS1c,CS1a,CS1b
BC4,BC1,BC2,CS1d,BC3,CS1c,CS1b,CS1a
BC4,BC1,BC3,CS1a,BC2,CS1b,CS1c,CS1d
BC4,BC1,BC3,CS1a,BC2,CS1b,CS1d,CS1c
BC4,BC1,BC3,CS1a,BC2,CS1c,CS1b,CS1d
BC4,BC1,BC3,CS1a,BC2,CS1c,CS1d,CS1b
BC4,BC1,BC3,CS1a,BC2,CS1d,CS1b,CS1c
BC4,BC1,BC3,CS1a,BC2,CS1d,CS1c,CS1b
BC4,BC1,BC3,CS1a,CS1c,BC2,CS1b,CS1d
BC4,BC1,BC3,CS1a,CS1c,BC2,CS1d,CS1b
BC4,BC1,BC3,CS1a,CS1c,CS1d,BC2,CS1b
BC4,BC1,BC3,CS1a,CS1d,BC2,CS1b,CS1c
BC4,BC1,BC3,CS1a,CS1d,BC2,CS1c,CS1b
BC4,BC1,BC3,CS1a,CS1d,CS1c,BC2,CS1b
BC4,BC1,BC3,BC2,CS1a,CS1b,CS1c,CS1d
BC4,BC1,BC3,BC2,CS1a,CS1b,CS1d,CS1c
BC4,BC1,BC3,BC2,CS1a,CS1c,CS1b,CS1d
BC4,BC1,BC3,BC2,CS1a,CS1c,CS1d,CS1b
BC4,BC1,BC3,BC2,CS1a,CS1d,CS1b,CS1c
BC4,BC1,BC3,BC2,CS1a,CS1d,CS1c,CS1b
BC4,BC1,BC3,BC2,CS1b,CS1a,CS1c,CS1d
BC4,BC1,BC3,BC2,CS1b,CS1a,CS1d,CS1c
BC4,BC1,BC3,BC2,CS1b,CS1c,CS1a,CS1d
BC4,BC1,BC3,BC2,CS1b,CS1c,CS1d,CS1a
BC4,BC1,BC3,BC2,CS1b,CS1d,CS1a,CS1c
BC4,BC1,BC3,BC2,CS1b,CS1d,CS1c,CS1a
BC4,BC1,BC3,BC2,CS1c,CS1a,CS1b,CS1d
BC4,BC1,BC3,BC2,CS1c,CS1a,CS1d,CS1b
BC4,BC1,BC3,BC2,CS1c,CS1b,CS1a,CS1d
BC4,BC1,BC3,BC2,CS1c,CS1b,CS1d,CS1a
BC4,BC1,BC3,BC2,CS1c,CS1d,CS1a,CS1b
BC4,BC1,BC3,BC2,CS1c,CS1d,CS1b,CS1a
BC4,BC1,BC3,BC2,CS1d,CS1a,CS1b,CS1c
BC4,BC1,BC3,BC2,CS1d,CS1a,CS1c,CS1b
BC4,BC1,BC3,BC2,CS1d,CS1b,CS1a,CS1c
BC4,BC1,BC3,BC2,CS1d,CS1b,CS1c,CS1a
BC4,BC1,BC3,BC2,CS1d,CS1c,CS1a,CS1b
BC4,BC1,BC3,BC2,CS1d,CS1c,CS1b,CS1a
BC4,BC1,BC3,CS1c,CS1a,BC2,CS1b,CS1d
BC4,BC1,BC3,CS1c,CS1a,BC2,CS1d,CS1b
BC4,BC1,BC3,CS1c,CS1a,CS1d,BC2,CS1b
BC4,BC1,BC3,CS1c,BC2,CS1a,CS1b,CS1d

BC4,BC1,BC3,CS1c,BC2,CS1a,CS1d,CS1b
BC4,BC1,BC3,CS1c,BC2,CS1b,CS1a,CS1d
BC4,BC1,BC3,CS1c,BC2,CS1b,CS1d,CS1a
BC4,BC1,BC3,CS1c,BC2,CS1d,CS1a,CS1b
BC4,BC1,BC3,CS1c,BC2,CS1d,CS1b,CS1a
BC4,BC1,BC3,CS1c,CS1d,CS1a,BC2,CS1b
BC4,BC1,BC3,CS1c,CS1d,BC2,CS1a,CS1b
BC4,BC1,BC3,CS1c,CS1d,BC2,CS1b,CS1a
BC4,BC1,BC3,CS1d,CS1a,BC2,CS1b,CS1c
BC4,BC1,BC3,CS1d,CS1a,BC2,CS1c,CS1b
BC4,BC1,BC3,CS1d,CS1a,CS1c,BC2,CS1b
BC4,BC1,BC3,CS1d,BC2,CS1a,CS1b,CS1c
BC4,BC1,BC3,CS1d,BC2,CS1a,CS1c,CS1b
BC4,BC1,BC3,CS1d,BC2,CS1b,CS1a,CS1c
BC4,BC1,BC3,CS1d,BC2,CS1b,CS1c,CS1a
BC4,BC1,BC3,CS1d,BC2,CS1c,CS1a,CS1b
BC4,BC1,BC3,CS1d,BC2,CS1c,CS1b,CS1a
BC4,BC1,BC3,CS1d,CS1c,CS1a,BC2,CS1b
BC4,BC1,BC3,CS1d,CS1c,BC2,CS1a,CS1b
BC4,BC1,BC3,CS1d,CS1c,BC2,CS1b,CS1a
BC4,BC1,CS1d,CS1a,BC2,CS1b,BC3,CS1c
BC4,BC1,CS1d,CS1a,BC2,BC3,CS1b,CS1c
BC4,BC1,CS1d,CS1a,BC2,BC3,CS1c,CS1b
BC4,BC1,CS1d,CS1a,BC3,BC2,CS1b,CS1c
BC4,BC1,CS1d,CS1a,BC3,BC2,CS1c,CS1b
BC4,BC1,CS1d,CS1a,BC3,CS1c,BC2,CS1b
BC4,BC1,CS1d,BC2,CS1a,CS1b,BC3,CS1c
BC4,BC1,CS1d,BC2,CS1a,BC3,CS1b,CS1c
BC4,BC1,CS1d,BC2,CS1a,BC3,CS1c,CS1b
BC4,BC1,CS1d,BC2,CS1b,CS1a,BC3,CS1c
BC4,BC1,CS1d,BC2,CS1b,BC3,CS1a,CS1c
BC4,BC1,CS1d,BC2,CS1b,BC3,CS1c,CS1a
BC4,BC1,CS1d,BC2,BC3,CS1a,CS1b,CS1c
BC4,BC1,CS1d,BC2,BC3,CS1a,CS1c,CS1b
BC4,BC1,CS1d,BC2,BC3,CS1b,CS1a,CS1c
BC4,BC1,CS1d,BC2,BC3,CS1b,CS1c,CS1a
BC4,BC1,CS1d,BC2,BC3,CS1c,CS1a,CS1b
BC4,BC1,CS1d,BC2,BC3,CS1c,CS1b,CS1a
BC4,BC1,CS1d,BC3,CS1a,BC2,CS1b,CS1c
BC4,BC1,CS1d,BC3,CS1a,BC2,CS1c,CS1b
BC4,BC1,CS1d,BC3,CS1a,CS1c,BC2,CS1b
BC4,BC1,CS1d,BC3,BC2,CS1a,CS1b,CS1c
BC4,BC1,CS1d,BC3,BC2,CS1a,CS1c,CS1b
BC4,BC1,CS1d,BC3,BC2,CS1b,CS1a,CS1c
BC4,BC1,CS1d,BC3,BC2,CS1b,CS1c,CS1a
BC4,BC1,CS1d,BC3,BC2,CS1c,CS1a,CS1b

BC4,BC1,CS1d,BC3,BC2,CS1c,CS1b,CS1a
BC4,BC1,CS1d,BC3,CS1c,CS1a,BC2,CS1b
BC4,BC1,CS1d,BC3,CS1c,BC2,CS1a,CS1b
BC4,BC1,CS1d,BC3,CS1c,BC2,CS1b,CS1a
BC4,BC2,BC1,CS1a,CS1b,BC3,CS1c,CS1d
BC4,BC2,BC1,CS1a,CS1b,BC3,CS1d,CS1c
BC4,BC2,BC1,CS1a,CS1b,CS1d,BC3,CS1c
BC4,BC2,BC1,CS1a,BC3,CS1b,CS1c,CS1d
BC4,BC2,BC1,CS1a,BC3,CS1b,CS1d,CS1c
BC4,BC2,BC1,CS1a,BC3,CS1c,CS1b,CS1d
BC4,BC2,BC1,CS1a,BC3,CS1c,CS1d,CS1b
BC4,BC2,BC1,CS1a,BC3,CS1d,CS1b,CS1c
BC4,BC2,BC1,CS1a,BC3,CS1d,CS1c,CS1b
BC4,BC2,BC1,CS1a,CS1d,CS1b,BC3,CS1c
BC4,BC2,BC1,CS1a,CS1d,BC3,CS1b,CS1c
BC4,BC2,BC1,CS1a,CS1d,BC3,CS1c,CS1b
BC4,BC2,BC1,CS1b,CS1a,BC3,CS1c,CS1d
BC4,BC2,BC1,CS1b,CS1a,BC3,CS1d,CS1c
BC4,BC2,BC1,CS1b,CS1a,CS1d,BC3,CS1c
BC4,BC2,BC1,CS1b,BC3,CS1a,CS1c,CS1d
BC4,BC2,BC1,CS1b,BC3,CS1a,CS1d,CS1c
BC4,BC2,BC1,CS1b,BC3,CS1c,CS1a,CS1d
BC4,BC2,BC1,CS1b,BC3,CS1c,CS1d,CS1a
BC4,BC2,BC1,CS1b,BC3,CS1d,CS1a,CS1c
BC4,BC2,BC1,CS1b,BC3,CS1d,CS1c,CS1a
BC4,BC2,BC1,CS1b,CS1d,CS1a,BC3,CS1c
BC4,BC2,BC1,CS1b,CS1d,BC3,CS1a,CS1c
BC4,BC2,BC1,CS1b,CS1d,BC3,CS1c,CS1a
BC4,BC2,BC1,BC3,CS1a,CS1b,CS1c,CS1d
BC4,BC2,BC1,BC3,CS1a,CS1b,CS1d,CS1c
BC4,BC2,BC1,BC3,CS1a,CS1c,CS1b,CS1d
BC4,BC2,BC1,BC3,CS1a,CS1c,CS1d,CS1b
BC4,BC2,BC1,BC3,CS1a,CS1d,CS1b,CS1c
BC4,BC2,BC1,BC3,CS1a,CS1d,CS1c,CS1b
BC4,BC2,BC1,BC3,CS1b,CS1a,CS1c,CS1d
BC4,BC2,BC1,BC3,CS1b,CS1a,CS1d,CS1c
BC4,BC2,BC1,BC3,CS1b,CS1c,CS1a,CS1d
BC4,BC2,BC1,BC3,CS1b,CS1c,CS1d,CS1a
BC4,BC2,BC1,BC3,CS1b,CS1d,CS1a,CS1c
BC4,BC2,BC1,BC3,CS1b,CS1d,CS1c,CS1a
BC4,BC2,BC1,BC3,CS1c,CS1a,CS1b,CS1d
BC4,BC2,BC1,BC3,CS1c,CS1a,CS1d,CS1b
BC4,BC2,BC1,BC3,CS1c,CS1b,CS1a,CS1d
BC4,BC2,BC1,BC3,CS1c,CS1b,CS1d,CS1a
BC4,BC2,BC1,BC3,CS1c,CS1d,CS1a,CS1b
BC4,BC2,BC1,BC3,CS1c,CS1d,CS1b,CS1a

BC4,BC2,BC1,BC3,CS1d,CS1a,CS1b,CS1c
BC4,BC2,BC1,BC3,CS1d,CS1a,CS1c,CS1b
BC4,BC2,BC1,BC3,CS1d,CS1b,CS1a,CS1c
BC4,BC2,BC1,BC3,CS1d,CS1b,CS1c,CS1a
BC4,BC2,BC1,BC3,CS1d,CS1c,CS1a,CS1b
BC4,BC2,BC1,BC3,CS1d,CS1c,CS1b,CS1a
BC4,BC2,BC1,CS1d,CS1a,CS1b,BC3,CS1c
BC4,BC2,BC1,CS1d,CS1a,BC3,CS1b,CS1c
BC4,BC2,BC1,CS1d,CS1a,BC3,CS1c,CS1b
BC4,BC2,BC1,CS1d,CS1b,CS1a,BC3,CS1c
BC4,BC2,BC1,CS1d,CS1b,BC3,CS1a,CS1c
BC4,BC2,BC1,CS1d,CS1b,BC3,CS1c,CS1a
BC4,BC2,BC1,CS1d,BC3,CS1a,CS1b,CS1c
BC4,BC2,BC1,CS1d,BC3,CS1a,CS1c,CS1b
BC4,BC2,BC1,CS1d,BC3,CS1b,CS1a,CS1c
BC4,BC2,BC1,CS1d,BC3,CS1b,CS1c,CS1a
BC4,BC2,BC1,CS1d,BC3,CS1c,CS1a,CS1b
BC4,BC2,BC1,CS1d,BC3,CS1c,CS1b,CS1a
BC4,BC2,CS1b,BC1,CS1a,BC3,CS1c,CS1d
BC4,BC2,CS1b,BC1,CS1a,BC3,CS1d,CS1c
BC4,BC2,CS1b,BC1,CS1a,CS1d,BC3,CS1c
BC4,BC2,CS1b,BC1,BC3,CS1a,CS1c,CS1d
BC4,BC2,CS1b,BC1,BC3,CS1a,CS1d,CS1c
BC4,BC2,CS1b,BC1,BC3,CS1c,CS1a,CS1d
BC4,BC2,CS1b,BC1,BC3,CS1c,CS1d,CS1a
BC4,BC2,CS1b,BC1,BC3,CS1d,CS1a,CS1c
BC4,BC2,CS1b,BC1,BC3,CS1d,CS1c,CS1a
BC4,BC2,CS1b,BC1,CS1d,CS1a,BC3,CS1c
BC4,BC2,CS1b,BC1,CS1d,BC3,CS1a,CS1c
BC4,BC2,CS1b,BC1,CS1d,BC3,CS1c,CS1a
BC4,BC2,CS1b,BC3,BC1,CS1a,CS1c,CS1d
BC4,BC2,CS1b,BC3,BC1,CS1a,CS1d,CS1c
BC4,BC2,CS1b,BC3,BC1,CS1c,CS1a,CS1d
BC4,BC2,CS1b,BC3,BC1,CS1c,CS1d,CS1a
BC4,BC2,CS1b,BC3,BC1,CS1d,CS1a,CS1c
BC4,BC2,CS1b,BC3,BC1,CS1d,CS1c,CS1a
BC4,BC2,CS1b,BC3,CS1c,BC1,CS1a,CS1d
BC4,BC2,CS1b,BC3,CS1c,BC1,CS1d,CS1a
BC4,BC2,CS1b,BC3,CS1c,CS1d,BC1,CS1a
BC4,BC2,CS1b,BC3,CS1d,BC1,CS1a,CS1c
BC4,BC2,CS1b,BC3,CS1d,BC1,CS1c,CS1a
BC4,BC2,CS1b,BC3,CS1d,CS1c,BC1,CS1a
BC4,BC2,CS1b,CS1d,BC1,CS1a,BC3,CS1c
BC4,BC2,CS1b,CS1d,BC1,BC3,CS1a,CS1c
BC4,BC2,CS1b,CS1d,BC1,BC3,CS1c,CS1a
BC4,BC2,CS1b,CS1d,BC3,BC1,CS1a,CS1c

BC4,BC2,CS1b,CS1d,BC3,BC1,CS1c,CS1a
BC4,BC2,CS1b,CS1d,BC3,CS1c,BC1,CS1a
BC4,BC2,BC3,BC1,CS1a,CS1b,CS1c,CS1d
BC4,BC2,BC3,BC1,CS1a,CS1b,CS1d,CS1c
BC4,BC2,BC3,BC1,CS1a,CS1c,CS1b,CS1d
BC4,BC2,BC3,BC1,CS1a,CS1c,CS1d,CS1b
BC4,BC2,BC3,BC1,CS1a,CS1d,CS1b,CS1c
BC4,BC2,BC3,BC1,CS1a,CS1d,CS1c,CS1b
BC4,BC2,BC3,BC1,CS1b,CS1a,CS1c,CS1d
BC4,BC2,BC3,BC1,CS1b,CS1a,CS1d,CS1c
BC4,BC2,BC3,BC1,CS1b,CS1c,CS1a,CS1d
BC4,BC2,BC3,BC1,CS1b,CS1c,CS1d,CS1a
BC4,BC2,BC3,BC1,CS1b,CS1d,CS1a,CS1c
BC4,BC2,BC3,BC1,CS1b,CS1d,CS1c,CS1a
BC4,BC2,BC3,BC1,CS1c,CS1a,CS1b,CS1d
BC4,BC2,BC3,BC1,CS1c,CS1a,CS1d,CS1b
BC4,BC2,BC3,BC1,CS1c,CS1b,CS1a,CS1d
BC4,BC2,BC3,BC1,CS1c,CS1b,CS1d,CS1a
BC4,BC2,BC3,BC1,CS1c,CS1d,CS1a,CS1b
BC4,BC2,BC3,BC1,CS1c,CS1d,CS1b,CS1a
BC4,BC2,BC3,BC1,CS1d,CS1a,CS1b,CS1c
BC4,BC2,BC3,BC1,CS1d,CS1a,CS1c,CS1b
BC4,BC2,BC3,BC1,CS1d,CS1b,CS1a,CS1c
BC4,BC2,BC3,BC1,CS1d,CS1b,CS1c,CS1a
BC4,BC2,BC3,BC1,CS1d,CS1c,CS1a,CS1b
BC4,BC2,BC3,BC1,CS1d,CS1c,CS1b,CS1a
BC4,BC2,BC3,CS1b,BC1,CS1a,CS1c,CS1d
BC4,BC2,BC3,CS1b,BC1,CS1a,CS1d,CS1c
BC4,BC2,BC3,CS1b,BC1,CS1c,CS1a,CS1d
BC4,BC2,BC3,CS1b,BC1,CS1c,CS1d,CS1a
BC4,BC2,BC3,CS1b,BC1,CS1d,CS1a,CS1c
BC4,BC2,BC3,CS1b,BC1,CS1d,CS1c,CS1a
BC4,BC2,BC3,CS1b,CS1c,BC1,CS1a,CS1d
BC4,BC2,BC3,CS1b,CS1c,BC1,CS1d,CS1a
BC4,BC2,BC3,CS1b,CS1c,CS1d,BC1,CS1a
BC4,BC2,BC3,CS1b,CS1d,BC1,CS1a,CS1c
BC4,BC2,BC3,CS1b,CS1d,BC1,CS1c,CS1a
BC4,BC2,BC3,CS1b,CS1d,CS1c,BC1,CS1a
BC4,BC2,BC3,CS1c,BC1,CS1a,CS1b,CS1d
BC4,BC2,BC3,CS1c,BC1,CS1a,CS1d,CS1b
BC4,BC2,BC3,CS1c,BC1,CS1b,CS1a,CS1d
BC4,BC2,BC3,CS1c,BC1,CS1b,CS1d,CS1a
BC4,BC2,BC3,CS1c,BC1,CS1d,CS1a,CS1b
BC4,BC2,BC3,CS1c,BC1,CS1d,CS1b,CS1a
BC4,BC2,BC3,CS1c,CS1b,BC1,CS1a,CS1d
BC4,BC2,BC3,CS1c,CS1b,BC1,CS1d,CS1a

BC4,BC2,BC3,CS1c,CS1b,CS1d,BC1,CS1a
BC4,BC2,BC3,CS1c,CS1d,BC1,CS1a,CS1b
BC4,BC2,BC3,CS1c,CS1d,BC1,CS1b,CS1a
BC4,BC2,BC3,CS1c,CS1d,CS1b,BC1,CS1a
BC4,BC2,BC3,CS1d,BC1,CS1a,CS1b,CS1c
BC4,BC2,BC3,CS1d,BC1,CS1a,CS1c,CS1b
BC4,BC2,BC3,CS1d,BC1,CS1b,CS1a,CS1c
BC4,BC2,BC3,CS1d,BC1,CS1b,CS1c,CS1a
BC4,BC2,BC3,CS1d,BC1,CS1c,CS1a,CS1b
BC4,BC2,BC3,CS1d,BC1,CS1c,CS1b,CS1a
BC4,BC2,BC3,CS1d,CS1b,BC1,CS1a,CS1c
BC4,BC2,BC3,CS1d,CS1b,BC1,CS1c,CS1a
BC4,BC2,BC3,CS1d,CS1b,CS1c,BC1,CS1a
BC4,BC2,BC3,CS1d,CS1c,BC1,CS1a,CS1b
BC4,BC2,BC3,CS1d,CS1c,BC1,CS1b,CS1a
BC4,BC2,BC3,CS1d,CS1c,CS1b,BC1,CS1a
BC4,BC2,CS1d,BC1,CS1a,CS1b,BC3,CS1c
BC4,BC2,CS1d,BC1,CS1a,BC3,CS1b,CS1c
BC4,BC2,CS1d,BC1,CS1a,BC3,CS1c,CS1b
BC4,BC2,CS1d,BC1,CS1b,CS1a,BC3,CS1c
BC4,BC2,CS1d,BC1,CS1b,BC3,CS1a,CS1c
BC4,BC2,CS1d,BC1,CS1b,BC3,CS1c,CS1a
BC4,BC2,CS1d,BC1,BC3,CS1a,CS1b,CS1c
BC4,BC2,CS1d,BC1,BC3,CS1a,CS1c,CS1b
BC4,BC2,CS1d,BC1,BC3,CS1b,CS1a,CS1c
BC4,BC2,CS1d,BC1,BC3,CS1b,CS1c,CS1a
BC4,BC2,CS1d,BC1,BC3,CS1c,CS1a,CS1b
BC4,BC2,CS1d,BC1,BC3,CS1c,CS1b,CS1a
BC4,BC2,CS1d,CS1b,BC1,CS1a,BC3,CS1c
BC4,BC2,CS1d,CS1b,BC1,BC3,CS1a,CS1c
BC4,BC2,CS1d,CS1b,BC1,BC3,CS1c,CS1a
BC4,BC2,CS1d,CS1b,BC3,BC1,CS1a,CS1c
BC4,BC2,CS1d,CS1b,BC3,BC1,CS1c,CS1a
BC4,BC2,CS1d,CS1b,BC3,CS1c,BC1,CS1a
BC4,BC2,CS1d,BC3,BC1,CS1a,CS1b,CS1c
BC4,BC2,CS1d,BC3,BC1,CS1a,CS1c,CS1b
BC4,BC2,CS1d,BC3,BC1,CS1b,CS1a,CS1c
BC4,BC2,CS1d,BC3,BC1,CS1b,CS1c,CS1a
BC4,BC2,CS1d,BC3,BC1,CS1c,CS1a,CS1b
BC4,BC2,CS1d,BC3,BC1,CS1c,CS1b,CS1a
BC4,BC2,CS1d,BC3,CS1b,BC1,CS1a,CS1c
BC4,BC2,CS1d,BC3,CS1b,BC1,CS1c,CS1a
BC4,BC2,CS1d,BC3,CS1b,CS1c,BC1,CS1a
BC4,BC2,CS1d,BC3,CS1c,BC1,CS1a,CS1b
BC4,BC2,CS1d,BC3,CS1c,BC1,CS1b,CS1a
BC4,BC2,CS1d,BC3,CS1c,CS1b,BC1,CS1a

BC4,BC3,BC1,CS1a,BC2,CS1b,CS1c,CS1d
BC4,BC3,BC1,CS1a,BC2,CS1b,CS1d,CS1c
BC4,BC3,BC1,CS1a,BC2,CS1c,CS1b,CS1d
BC4,BC3,BC1,CS1a,BC2,CS1c,CS1d,CS1b
BC4,BC3,BC1,CS1a,BC2,CS1d,CS1b,CS1c
BC4,BC3,BC1,CS1a,BC2,CS1d,CS1c,CS1b
BC4,BC3,BC1,CS1a,CS1c,BC2,CS1b,CS1d
BC4,BC3,BC1,CS1a,CS1c,BC2,CS1d,CS1b
BC4,BC3,BC1,CS1a,CS1c,CS1d,BC2,CS1b
BC4,BC3,BC1,CS1a,CS1d,BC2,CS1b,CS1c
BC4,BC3,BC1,CS1a,CS1d,BC2,CS1c,CS1b
BC4,BC3,BC1,CS1a,CS1d,CS1c,BC2,CS1b
BC4,BC3,BC1,BC2,CS1a,CS1b,CS1c,CS1d
BC4,BC3,BC1,BC2,CS1a,CS1b,CS1d,CS1c
BC4,BC3,BC1,BC2,CS1a,CS1c,CS1b,CS1d
BC4,BC3,BC1,BC2,CS1a,CS1c,CS1d,CS1b
BC4,BC3,BC1,BC2,CS1a,CS1d,CS1b,CS1c
BC4,BC3,BC1,BC2,CS1a,CS1d,CS1c,CS1b
BC4,BC3,BC1,BC2,CS1b,CS1a,CS1c,CS1d
BC4,BC3,BC1,BC2,CS1b,CS1a,CS1d,CS1c
BC4,BC3,BC1,BC2,CS1b,CS1c,CS1a,CS1d
BC4,BC3,BC1,BC2,CS1b,CS1c,CS1d,CS1a
BC4,BC3,BC1,BC2,CS1b,CS1d,CS1a,CS1c
BC4,BC3,BC1,BC2,CS1b,CS1d,CS1c,CS1a
BC4,BC3,BC1,BC2,CS1c,CS1a,CS1b,CS1d
BC4,BC3,BC1,BC2,CS1c,CS1a,CS1d,CS1b
BC4,BC3,BC1,BC2,CS1c,CS1b,CS1a,CS1d
BC4,BC3,BC1,BC2,CS1c,CS1b,CS1d,CS1a
BC4,BC3,BC1,BC2,CS1c,CS1d,CS1a,CS1b
BC4,BC3,BC1,BC2,CS1c,CS1d,CS1b,CS1a
BC4,BC3,BC1,BC2,CS1d,CS1a,CS1b,CS1c
BC4,BC3,BC1,BC2,CS1d,CS1a,CS1c,CS1b
BC4,BC3,BC1,BC2,CS1d,CS1b,CS1a,CS1c
BC4,BC3,BC1,BC2,CS1d,CS1b,CS1c,CS1a
BC4,BC3,BC1,BC2,CS1d,CS1c,CS1a,CS1b
BC4,BC3,BC1,BC2,CS1d,CS1c,CS1b,CS1a
BC4,BC3,BC1,CS1c,CS1a,BC2,CS1b,CS1d
BC4,BC3,BC1,CS1c,CS1a,BC2,CS1d,CS1b
BC4,BC3,BC1,CS1c,CS1a,CS1d,BC2,CS1b
BC4,BC3,BC1,CS1c,BC2,CS1a,CS1b,CS1d
BC4,BC3,BC1,CS1c,BC2,CS1a,CS1d,CS1b
BC4,BC3,BC1,CS1c,BC2,CS1b,CS1a,CS1d
BC4,BC3,BC1,CS1c,BC2,CS1b,CS1d,CS1a
BC4,BC3,BC1,CS1c,BC2,CS1d,CS1a,CS1b
BC4,BC3,BC1,CS1c,BC2,CS1d,CS1b,CS1a
BC4,BC3,BC1,CS1c,CS1d,CS1a,BC2,CS1b

BC4,BC3,BC1,CS1c,CS1d,BC2,CS1a,CS1b
BC4,BC3,BC1,CS1c,CS1d,BC2,CS1b,CS1a
BC4,BC3,BC1,CS1d,CS1a,BC2,CS1b,CS1c
BC4,BC3,BC1,CS1d,CS1a,BC2,CS1c,CS1b
BC4,BC3,BC1,CS1d,CS1a,CS1c,BC2,CS1b
BC4,BC3,BC1,CS1d,BC2,CS1a,CS1b,CS1c
BC4,BC3,BC1,CS1d,BC2,CS1a,CS1c,CS1b
BC4,BC3,BC1,CS1d,BC2,CS1b,CS1a,CS1c
BC4,BC3,BC1,CS1d,BC2,CS1b,CS1c,CS1a
BC4,BC3,BC1,CS1d,BC2,CS1c,CS1a,CS1b
BC4,BC3,BC1,CS1d,BC2,CS1c,CS1b,CS1a
BC4,BC3,BC1,CS1d,CS1c,CS1a,BC2,CS1b
BC4,BC3,BC1,CS1d,CS1c,BC2,CS1a,CS1b
BC4,BC3,BC1,CS1d,CS1c,BC2,CS1b,CS1a
BC4,BC3,BC2,BC1,CS1a,CS1b,CS1c,CS1d
BC4,BC3,BC2,BC1,CS1a,CS1b,CS1d,CS1c
BC4,BC3,BC2,BC1,CS1a,CS1c,CS1b,CS1d
BC4,BC3,BC2,BC1,CS1a,CS1c,CS1d,CS1b
BC4,BC3,BC2,BC1,CS1a,CS1d,CS1b,CS1c
BC4,BC3,BC2,BC1,CS1a,CS1d,CS1c,CS1b
BC4,BC3,BC2,BC1,CS1b,CS1a,CS1c,CS1d
BC4,BC3,BC2,BC1,CS1b,CS1a,CS1d,CS1c
BC4,BC3,BC2,BC1,CS1b,CS1c,CS1a,CS1d
BC4,BC3,BC2,BC1,CS1b,CS1c,CS1d,CS1a
BC4,BC3,BC2,BC1,CS1b,CS1d,CS1a,CS1c
BC4,BC3,BC2,BC1,CS1b,CS1d,CS1c,CS1a
BC4,BC3,BC2,BC1,CS1c,CS1a,CS1b,CS1d
BC4,BC3,BC2,BC1,CS1c,CS1a,CS1d,CS1b
BC4,BC3,BC2,BC1,CS1c,CS1b,CS1a,CS1d
BC4,BC3,BC2,BC1,CS1c,CS1b,CS1d,CS1a
BC4,BC3,BC2,BC1,CS1c,CS1d,CS1a,CS1b
BC4,BC3,BC2,BC1,CS1c,CS1d,CS1b,CS1a
BC4,BC3,BC2,BC1,CS1d,CS1a,CS1b,CS1c
BC4,BC3,BC2,BC1,CS1d,CS1a,CS1c,CS1b
BC4,BC3,BC2,BC1,CS1d,CS1b,CS1a,CS1c
BC4,BC3,BC2,BC1,CS1d,CS1b,CS1c,CS1a
BC4,BC3,BC2,BC1,CS1d,CS1c,CS1a,CS1b
BC4,BC3,BC2,BC1,CS1d,CS1c,CS1b,CS1a
BC4,BC3,BC2,CS1b,BC1,CS1a,CS1c,CS1d
BC4,BC3,BC2,CS1b,BC1,CS1a,CS1d,CS1c
BC4,BC3,BC2,CS1b,BC1,CS1c,CS1a,CS1d
BC4,BC3,BC2,CS1b,BC1,CS1c,CS1d,CS1a
BC4,BC3,BC2,CS1b,BC1,CS1d,CS1a,CS1c
BC4,BC3,BC2,CS1b,BC1,CS1d,CS1c,CS1a
BC4,BC3,BC2,CS1b,CS1c,BC1,CS1a,CS1d
BC4,BC3,BC2,CS1b,CS1c,BC1,CS1d,CS1a

BC4,BC3,BC2,CS1b,CS1c,CS1d,BC1,CS1a
BC4,BC3,BC2,CS1b,CS1d,BC1,CS1a,CS1c
BC4,BC3,BC2,CS1b,CS1d,BC1,CS1c,CS1a
BC4,BC3,BC2,CS1b,CS1d,CS1c,BC1,CS1a
BC4,BC3,BC2,CS1c,BC1,CS1a,CS1b,CS1d
BC4,BC3,BC2,CS1c,BC1,CS1a,CS1d,CS1b
BC4,BC3,BC2,CS1c,BC1,CS1b,CS1a,CS1d
BC4,BC3,BC2,CS1c,BC1,CS1b,CS1d,CS1a
BC4,BC3,BC2,CS1c,BC1,CS1d,CS1a,CS1b
BC4,BC3,BC2,CS1c,BC1,CS1d,CS1b,CS1a
BC4,BC3,BC2,CS1c,CS1b,BC1,CS1a,CS1d
BC4,BC3,BC2,CS1c,CS1b,BC1,CS1d,CS1a
BC4,BC3,BC2,CS1c,CS1b,CS1d,BC1,CS1a
BC4,BC3,BC2,CS1c,CS1d,BC1,CS1a,CS1b
BC4,BC3,BC2,CS1c,CS1d,BC1,CS1b,CS1a
BC4,BC3,BC2,CS1c,CS1d,CS1b,BC1,CS1a
BC4,BC3,BC2,CS1d,BC1,CS1a,CS1b,CS1c
BC4,BC3,BC2,CS1d,BC1,CS1a,CS1c,CS1b
BC4,BC3,BC2,CS1d,BC1,CS1b,CS1a,CS1c
BC4,BC3,BC2,CS1d,BC1,CS1b,CS1c,CS1a
BC4,BC3,BC2,CS1d,BC1,CS1c,CS1a,CS1b
BC4,BC3,BC2,CS1d,BC1,CS1c,CS1b,CS1a
BC4,BC3,BC2,CS1d,CS1b,BC1,CS1a,CS1c
BC4,BC3,BC2,CS1d,CS1b,BC1,CS1c,CS1a
BC4,BC3,BC2,CS1d,CS1b,CS1c,BC1,CS1a
BC4,BC3,BC2,CS1d,CS1c,BC1,CS1a,CS1b
BC4,BC3,BC2,CS1d,CS1c,BC1,CS1b,CS1a
BC4,BC3,BC2,CS1d,CS1c,CS1b,BC1,CS1a
BC4,BC3,CS1c,BC1,CS1a,BC2,CS1b,CS1d
BC4,BC3,CS1c,BC1,CS1a,BC2,CS1d,CS1b
BC4,BC3,CS1c,BC1,CS1a,CS1d,BC2,CS1b
BC4,BC3,CS1c,BC1,BC2,CS1a,CS1b,CS1d
BC4,BC3,CS1c,BC1,BC2,CS1a,CS1d,CS1b
BC4,BC3,CS1c,BC1,BC2,CS1b,CS1a,CS1d
BC4,BC3,CS1c,BC1,BC2,CS1b,CS1d,CS1a
BC4,BC3,CS1c,BC1,BC2,CS1d,CS1a,CS1b
BC4,BC3,CS1c,BC1,BC2,CS1d,CS1b,CS1a
BC4,BC3,CS1c,BC1,CS1d,CS1a,BC2,CS1b
BC4,BC3,CS1c,BC1,CS1d,BC2,CS1a,CS1b
BC4,BC3,CS1c,BC1,CS1d,BC2,CS1b,CS1a
BC4,BC3,CS1c,BC2,BC1,CS1a,CS1b,CS1d
BC4,BC3,CS1c,BC2,BC1,CS1a,CS1d,CS1b
BC4,BC3,CS1c,BC2,BC1,CS1b,CS1a,CS1d
BC4,BC3,CS1c,BC2,BC1,CS1b,CS1d,CS1a
BC4,BC3,CS1c,BC2,BC1,CS1d,CS1a,CS1b
BC4,BC3,CS1c,BC2,BC1,CS1d,CS1b,CS1a

BC4,BC3,CS1c,BC2,CS1b,BC1,CS1a,CS1d
BC4,BC3,CS1c,BC2,CS1b,BC1,CS1d,CS1a
BC4,BC3,CS1c,BC2,CS1b,CS1d,BC1,CS1a
BC4,BC3,CS1c,BC2,CS1d,BC1,CS1a,CS1b
BC4,BC3,CS1c,BC2,CS1d,BC1,CS1b,CS1a
BC4,BC3,CS1c,BC2,CS1d,CS1b,BC1,CS1a
BC4,BC3,CS1c,CS1d,BC1,CS1a,BC2,CS1b
BC4,BC3,CS1c,CS1d,BC1,BC2,CS1a,CS1b
BC4,BC3,CS1c,CS1d,BC1,BC2,CS1b,CS1a
BC4,BC3,CS1c,CS1d,BC2,BC1,CS1a,CS1b
BC4,BC3,CS1c,CS1d,BC2,BC1,CS1b,CS1a
BC4,BC3,CS1c,CS1d,BC2,CS1b,BC1,CS1a
BC4,BC3,CS1d,BC1,CS1a,BC2,CS1b,CS1c
BC4,BC3,CS1d,BC1,CS1a,BC2,CS1c,CS1b
BC4,BC3,CS1d,BC1,CS1a,CS1c,BC2,CS1b
BC4,BC3,CS1d,BC1,BC2,CS1a,CS1b,CS1c
BC4,BC3,CS1d,BC1,BC2,CS1a,CS1c,CS1b
BC4,BC3,CS1d,BC1,BC2,CS1b,CS1a,CS1c
BC4,BC3,CS1d,BC1,BC2,CS1b,CS1c,CS1a
BC4,BC3,CS1d,BC1,BC2,CS1c,CS1a,CS1b
BC4,BC3,CS1d,BC1,BC2,CS1c,CS1b,CS1a
BC4,BC3,CS1d,BC1,CS1c,CS1a,BC2,CS1b
BC4,BC3,CS1d,BC1,CS1c,BC2,CS1a,CS1b
BC4,BC3,CS1d,BC1,CS1c,BC2,CS1b,CS1a
BC4,BC3,CS1d,BC2,BC1,CS1a,CS1b,CS1c
BC4,BC3,CS1d,BC2,BC1,CS1a,CS1c,CS1b
BC4,BC3,CS1d,BC2,BC1,CS1b,CS1a,CS1c
BC4,BC3,CS1d,BC2,BC1,CS1b,CS1c,CS1a
BC4,BC3,CS1d,BC2,BC1,CS1c,CS1a,CS1b
BC4,BC3,CS1d,BC2,BC1,CS1c,CS1b,CS1a
BC4,BC3,CS1d,BC2,CS1b,BC1,CS1a,CS1c
BC4,BC3,CS1d,BC2,CS1b,BC1,CS1c,CS1a
BC4,BC3,CS1d,BC2,CS1b,CS1c,BC1,CS1a
BC4,BC3,CS1d,BC2,CS1c,BC1,CS1a,CS1b
BC4,BC3,CS1d,BC2,CS1c,BC1,CS1b,CS1a
BC4,BC3,CS1d,BC2,CS1c,CS1b,BC1,CS1a
BC4,BC3,CS1d,CS1c,BC1,CS1a,BC2,CS1b
BC4,BC3,CS1d,CS1c,BC1,BC2,CS1a,CS1b
BC4,BC3,CS1d,CS1c,BC1,BC2,CS1b,CS1a
BC4,BC3,CS1d,CS1c,BC2,BC1,CS1a,CS1b
BC4,BC3,CS1d,CS1c,BC2,BC1,CS1b,CS1a
BC4,BC3,CS1d,CS1c,BC2,CS1b,BC1,CS1a
BC4,CS1d,BC1,CS1a,BC2,CS1b,BC3,CS1c
BC4,CS1d,BC1,CS1a,BC2,BC3,CS1b,CS1c
BC4,CS1d,BC1,CS1a,BC2,BC3,CS1c,CS1b
BC4,CS1d,BC1,CS1a,BC3,BC2,CS1b,CS1c

BC4,CS1d,BC1,CS1a,BC3,BC2,CS1c,CS1b
BC4,CS1d,BC1,CS1a,BC3,CS1c,BC2,CS1b
BC4,CS1d,BC1,BC2,CS1a,CS1b,BC3,CS1c
BC4,CS1d,BC1,BC2,CS1a,BC3,CS1b,CS1c
BC4,CS1d,BC1,BC2,CS1a,BC3,CS1c,CS1b
BC4,CS1d,BC1,BC2,CS1b,CS1a,BC3,CS1c
BC4,CS1d,BC1,BC2,CS1b,BC3,CS1a,CS1c
BC4,CS1d,BC1,BC2,CS1b,BC3,CS1c,CS1a
BC4,CS1d,BC1,BC2,BC3,CS1a,CS1b,CS1c
BC4,CS1d,BC1,BC2,BC3,CS1a,CS1c,CS1b
BC4,CS1d,BC1,BC2,BC3,CS1b,CS1a,CS1c
BC4,CS1d,BC1,BC2,BC3,CS1b,CS1c,CS1a
BC4,CS1d,BC1,BC2,BC3,CS1c,CS1a,CS1b
BC4,CS1d,BC1,BC2,BC3,CS1c,CS1b,CS1a
BC4,CS1d,BC1,BC3,CS1a,BC2,CS1b,CS1c
BC4,CS1d,BC1,BC3,CS1a,BC2,CS1c,CS1b
BC4,CS1d,BC1,BC3,CS1a,CS1c,BC2,CS1b
BC4,CS1d,BC1,BC3,BC2,CS1a,CS1b,CS1c
BC4,CS1d,BC1,BC3,BC2,CS1a,CS1c,CS1b
BC4,CS1d,BC1,BC3,BC2,CS1b,CS1a,CS1c
BC4,CS1d,BC1,BC3,BC2,CS1b,CS1c,CS1a
BC4,CS1d,BC1,BC3,BC2,CS1c,CS1a,CS1b
BC4,CS1d,BC1,BC3,BC2,CS1c,CS1b,CS1a
BC4,CS1d,BC1,BC3,CS1c,CS1a,BC2,CS1b
BC4,CS1d,BC1,BC3,CS1c,BC2,CS1a,CS1b
BC4,CS1d,BC1,BC3,CS1c,BC2,CS1b,CS1a
BC4,CS1d,BC2,BC1,CS1a,CS1b,BC3,CS1c
BC4,CS1d,BC2,BC1,CS1a,BC3,CS1b,CS1c
BC4,CS1d,BC2,BC1,CS1a,BC3,CS1c,CS1b
BC4,CS1d,BC2,BC1,CS1b,CS1a,BC3,CS1c
BC4,CS1d,BC2,BC1,CS1b,BC3,CS1a,CS1c
BC4,CS1d,BC2,BC1,CS1b,BC3,CS1c,CS1a
BC4,CS1d,BC2,BC1,BC3,CS1a,CS1b,CS1c
BC4,CS1d,BC2,BC1,BC3,CS1a,CS1c,CS1b
BC4,CS1d,BC2,BC1,BC3,CS1b,CS1a,CS1c
BC4,CS1d,BC2,BC1,BC3,CS1b,CS1c,CS1a
BC4,CS1d,BC2,BC1,BC3,CS1c,CS1a,CS1b
BC4,CS1d,BC2,BC1,BC3,CS1c,CS1b,CS1a
BC4,CS1d,BC2,CS1b,BC1,CS1a,BC3,CS1c
BC4,CS1d,BC2,CS1b,BC1,BC3,CS1a,CS1c
BC4,CS1d,BC2,CS1b,BC1,BC3,CS1c,CS1a
BC4,CS1d,BC2,CS1b,BC3,BC1,CS1a,CS1c
BC4,CS1d,BC2,CS1b,BC3,BC1,CS1c,CS1a
BC4,CS1d,BC2,CS1b,BC3,CS1c,BC1,CS1a
BC4,CS1d,BC2,BC3,BC1,CS1a,CS1b,CS1c
BC4,CS1d,BC2,BC3,BC1,CS1a,CS1c,CS1b

BC4,CS1d,BC2,BC3,BC1,CS1b,CS1a,CS1c
BC4,CS1d,BC2,BC3,BC1,CS1b,CS1c,CS1a
BC4,CS1d,BC2,BC3,BC1,CS1c,CS1a,CS1b
BC4,CS1d,BC2,BC3,BC1,CS1c,CS1b,CS1a
BC4,CS1d,BC2,BC3,CS1b,BC1,CS1a,CS1c
BC4,CS1d,BC2,BC3,CS1b,BC1,CS1c,CS1a
BC4,CS1d,BC2,BC3,CS1b,CS1c,BC1,CS1a
BC4,CS1d,BC2,BC3,CS1c,BC1,CS1a,CS1b
BC4,CS1d,BC2,BC3,CS1c,BC1,CS1b,CS1a
BC4,CS1d,BC2,BC3,CS1c,CS1b,BC1,CS1a
BC4,CS1d,BC3,BC1,CS1a,BC2,CS1b,CS1c
BC4,CS1d,BC3,BC1,CS1a,BC2,CS1c,CS1b
BC4,CS1d,BC3,BC1,CS1a,CS1c,BC2,CS1b
BC4,CS1d,BC3,BC1,BC2,CS1a,CS1b,CS1c
BC4,CS1d,BC3,BC1,BC2,CS1a,CS1c,CS1b
BC4,CS1d,BC3,BC1,BC2,CS1b,CS1a,CS1c
BC4,CS1d,BC3,BC1,BC2,CS1b,CS1c,CS1a
BC4,CS1d,BC3,BC1,BC2,CS1c,CS1a,CS1b
BC4,CS1d,BC3,BC1,BC2,CS1c,CS1b,CS1a
BC4,CS1d,BC3,BC1,CS1c,CS1a,BC2,CS1b
BC4,CS1d,BC3,BC1,CS1c,BC2,CS1a,CS1b
BC4,CS1d,BC3,BC1,CS1c,BC2,CS1b,CS1a
BC4,CS1d,BC3,BC2,BC1,CS1a,CS1b,CS1c
BC4,CS1d,BC3,BC2,BC1,CS1a,CS1c,CS1b
BC4,CS1d,BC3,BC2,BC1,CS1b,CS1a,CS1c
BC4,CS1d,BC3,BC2,BC1,CS1b,CS1c,CS1a
BC4,CS1d,BC3,BC2,BC1,CS1c,CS1a,CS1b
BC4,CS1d,BC3,BC2,BC1,CS1c,CS1b,CS1a
BC4,CS1d,BC3,BC2,CS1b,BC1,CS1a,CS1c
BC4,CS1d,BC3,BC2,CS1b,BC1,CS1c,CS1a
BC4,CS1d,BC3,BC2,CS1b,CS1c,BC1,CS1a
BC4,CS1d,BC3,BC2,CS1c,BC1,CS1a,CS1b
BC4,CS1d,BC3,BC2,CS1c,BC1,CS1b,CS1a
BC4,CS1d,BC3,BC2,CS1c,CS1b,BC1,CS1a
BC4,CS1d,BC3,CS1c,BC1,CS1a,BC2,CS1b
BC4,CS1d,BC3,CS1c,BC1,BC2,CS1a,CS1b
BC4,CS1d,BC3,CS1c,BC1,BC2,CS1b,CS1a
BC4,CS1d,BC3,CS1c,BC2,BC1,CS1a,CS1b
BC4,CS1d,BC3,CS1c,BC2,BC1,CS1b,CS1a
BC4,CS1d,BC3,CS1c,BC2,CS1b,BC1,CS1a

Appendix J

Equivalent classes for the results for Case 5

BC1,CS1a,BC2,CS1b,BC3,CS1c,BC4,CS1d
BC1,CS1a,BC2,CS1b,BC3,BC4,CS1c,CS1d
BC1,CS1a,BC2,CS1b,BC3,BC4,CS1d,CS1c
BC1,CS1a,BC2,CS1b,BC4,CS1d,BC3,CS1c
BC1,CS1a,BC2,BC3,CS1b,CS1c,BC4,CS1d
BC1,CS1a,BC2,BC3,CS1b,BC4,CS1c,CS1d
BC1,CS1a,BC2,BC3,CS1b,BC4,CS1d,CS1c
BC1,CS1a,BC2,BC3,CS1c,CS1b,BC4,CS1d
BC1,CS1a,BC2,BC3,CS1c,BC4,CS1b,CS1d
BC1,CS1a,BC2,BC3,CS1c,BC4,CS1d,CS1b
BC1,CS1a,BC2,BC3,BC4,CS1b,CS1c,CS1d
BC1,CS1a,BC2,BC3,BC4,CS1b,CS1d,CS1c
BC1,CS1a,BC2,BC3,BC4,CS1c,CS1b,CS1d
BC1,CS1a,BC2,BC3,BC4,CS1c,CS1d,CS1b
BC1,CS1a,BC2,BC3,BC4,CS1d,CS1b,CS1c
BC1,CS1a,BC2,BC3,BC4,CS1d,CS1c,CS1b
BC1,CS1a,BC2,BC4,CS1b,BC3,CS1c,CS1d
BC1,CS1a,BC2,BC4,CS1b,BC3,CS1d,CS1c
BC1,CS1a,BC2,BC4,CS1b,CS1d,BC3,CS1c
BC1,CS1a,BC2,BC4,CS1d,BC3,CS1b,CS1c
BC1,CS1a,BC2,BC4,CS1d,BC3,CS1c,CS1b
BC1,CS1a,BC3,CS1c,BC2,CS1b,BC4,CS1d
BC1,CS1a,BC3,CS1c,BC2,BC4,CS1b,CS1d
BC1,CS1a,BC3,CS1c,BC2,BC4,CS1d,CS1b
BC1,CS1a,BC3,CS1c,BC4,CS1d,BC2,CS1b
BC1,CS1a,BC3,BC4,CS1c,BC2,CS1b,CS1d
BC1,CS1a,BC3,BC4,CS1c,BC2,CS1d,CS1b
BC1,CS1a,BC3,BC4,CS1c,CS1d,BC2,CS1b
BC1,CS1a,BC3,BC4,CS1d,BC2,CS1b,CS1c
BC1,CS1a,BC3,BC4,CS1d,BC2,CS1c,CS1b
BC1,CS1a,BC3,BC4,CS1d,CS1c,BC2,CS1b
BC1,CS1a,BC4,CS1d,BC2,CS1b,BC3,CS1c
BC1,CS1a,BC4,CS1d,BC2,BC3,CS1b,CS1c
BC1,CS1a,BC4,CS1d,BC2,BC3,CS1c,CS1b
BC1,CS1a,BC4,CS1d,BC3,CS1c,BC2,CS1b
BC1,BC2,CS1b,CS1a,BC3,CS1c,BC4,CS1d
BC1,BC2,CS1b,CS1a,BC3,BC4,CS1c,CS1d
BC1,BC2,CS1b,CS1a,BC3,BC4,CS1d,CS1c
BC1,BC2,CS1b,CS1a,BC4,CS1d,BC3,CS1c
BC1,BC2,CS1b,BC3,CS1a,CS1c,BC4,CS1d
BC1,BC2,CS1b,BC3,CS1a,BC4,CS1c,CS1d
BC1,BC2,CS1b,BC3,CS1a,BC4,CS1d,CS1c

BC1,BC2,CS1b,BC3,CS1c,CS1a,BC4,CS1d
BC1,BC2,CS1b,BC3,CS1c,BC4,CS1a,CS1d
BC1,BC2,CS1b,BC3,CS1c,BC4,CS1d,CS1a
BC1,BC2,CS1b,BC3,BC4,CS1a,CS1c,CS1d
BC1,BC2,CS1b,BC3,BC4,CS1a,CS1d,CS1c
BC1,BC2,CS1b,BC3,BC4,CS1c,CS1a,CS1d
BC1,BC2,CS1b,BC3,BC4,CS1c,CS1d,CS1a
BC1,BC2,CS1b,BC3,BC4,CS1d,CS1a,CS1c
BC1,BC2,CS1b,BC3,BC4,CS1d,CS1c,CS1a
BC1,BC2,CS1b,BC4,CS1a,BC3,CS1c,CS1d
BC1,BC2,CS1b,BC4,CS1a,BC3,CS1d,CS1c
BC1,BC2,CS1b,BC4,CS1a,CS1d,BC3,CS1c
BC1,BC2,CS1b,BC4,CS1d,CS1a,BC3,CS1c
BC1,BC2,CS1b,BC4,CS1d,BC3,CS1a,CS1c
BC1,BC2,CS1b,BC4,CS1d,BC3,CS1c,CS1a
BC1,BC2,BC3,CS1c,CS1a,CS1b,BC4,CS1d
BC1,BC2,BC3,CS1c,CS1a,BC4,CS1b,CS1d
BC1,BC2,BC3,CS1c,CS1a,BC4,CS1d,CS1b
BC1,BC2,BC3,CS1c,CS1b,CS1a,BC4,CS1d
BC1,BC2,BC3,CS1c,CS1b,BC4,CS1a,CS1d
BC1,BC2,BC3,CS1c,CS1b,BC4,CS1d,CS1a
BC1,BC2,BC3,CS1c,BC4,CS1a,CS1b,CS1d
BC1,BC2,BC3,CS1c,BC4,CS1a,CS1d,CS1b
BC1,BC2,BC3,CS1c,BC4,CS1b,CS1a,CS1d
BC1,BC2,BC3,CS1c,BC4,CS1b,CS1d,CS1a
BC1,BC2,BC3,CS1c,BC4,CS1d,CS1a,CS1b
BC1,BC2,BC3,CS1c,BC4,CS1d,CS1b,CS1a
BC1,BC2,BC3,BC4,CS1d,CS1a,CS1b,CS1c
BC1,BC2,BC3,BC4,CS1d,CS1a,CS1c,CS1b
BC1,BC2,BC3,BC4,CS1d,CS1b,CS1a,CS1c
BC1,BC2,BC3,BC4,CS1d,CS1b,CS1c,CS1a
BC1,BC2,BC3,BC4,CS1d,CS1c,CS1a,CS1b
BC1,BC2,BC3,BC4,CS1d,CS1c,CS1b,CS1a
BC1,BC2,BC4,CS1d,CS1a,CS1b,BC3,CS1c
BC1,BC2,BC4,CS1d,CS1a,BC3,CS1b,CS1c
BC1,BC2,BC4,CS1d,CS1a,BC3,CS1c,CS1b
BC1,BC2,BC4,CS1d,CS1b,CS1a,BC3,CS1c
BC1,BC2,BC4,CS1d,CS1b,BC3,CS1a,CS1c
BC1,BC2,BC4,CS1d,CS1b,BC3,CS1c,CS1a
BC1,BC3,CS1c,CS1a,BC2,CS1b,BC4,CS1d
BC1,BC3,CS1c,CS1a,BC2,BC4,CS1b,CS1d
BC1,BC3,CS1c,CS1a,BC2,BC4,CS1d,CS1b
BC1,BC3,CS1c,CS1a,BC4,CS1d,BC2,CS1b
BC1,BC3,CS1c,BC4,CS1a,BC2,CS1b,CS1d
BC1,BC3,CS1c,BC4,CS1a,BC2,CS1d,CS1b
BC1,BC3,CS1c,BC4,CS1a,CS1d,BC2,CS1b

BC1,BC3,CS1c,BC4,CS1d,CS1a,BC2,CS1b
BC1,BC3,CS1c,BC4,CS1d,BC2,CS1a,CS1b
BC1,BC3,CS1c,BC4,CS1d,BC2,CS1b,CS1a
BC1,BC3,BC4,CS1d,CS1a,BC2,CS1b,CS1c
BC1,BC3,BC4,CS1d,CS1a,BC2,CS1c,CS1b
BC1,BC3,BC4,CS1d,CS1a,CS1c,BC2,CS1b
BC1,BC3,BC4,CS1d,CS1c,CS1a,BC2,CS1b
BC1,BC3,BC4,CS1d,CS1c,BC2,CS1a,CS1b
BC1,BC3,BC4,CS1d,CS1c,BC2,CS1b,CS1a
BC1,BC4,CS1d,CS1a,BC2,CS1b,BC3,CS1c
BC1,BC4,CS1d,CS1a,BC2,BC3,CS1b,CS1c
BC1,BC4,CS1d,CS1a,BC2,BC3,CS1c,CS1b
BC1,BC4,CS1d,CS1a,BC3,CS1c,BC2,CS1b
BC2,CS1b,BC3,CS1c,BC1,CS1a,BC4,CS1d
BC2,CS1b,BC3,CS1c,BC1,BC4,CS1a,CS1d
BC2,CS1b,BC3,CS1c,BC1,BC4,CS1d,CS1a
BC2,CS1b,BC3,CS1c,BC4,CS1d,BC1,CS1a
BC2,CS1b,BC3,BC4,CS1c,BC1,CS1a,CS1d
BC2,CS1b,BC3,BC4,CS1c,BC1,CS1d,CS1a
BC2,CS1b,BC3,BC4,CS1c,CS1d,BC1,CS1a
BC2,CS1b,BC3,BC4,CS1d,BC1,CS1a,CS1c
BC2,CS1b,BC3,BC4,CS1d,BC1,CS1c,CS1a
BC2,CS1b,BC3,BC4,CS1d,CS1c,BC1,CS1a
BC2,CS1b,BC4,CS1d,BC1,CS1a,BC3,CS1c
BC2,CS1b,BC4,CS1d,BC1,BC3,CS1a,CS1c
BC2,CS1b,BC4,CS1d,BC1,BC3,CS1c,CS1a
BC2,CS1b,BC4,CS1d,BC3,CS1c,BC1,CS1a
BC2,BC3,CS1c,CS1b,BC1,CS1a,BC4,CS1d
BC2,BC3,CS1c,CS1b,BC1,BC4,CS1a,CS1d
BC2,BC3,CS1c,CS1b,BC1,BC4,CS1d,CS1a
BC2,BC3,CS1c,CS1b,BC4,CS1d,BC1,CS1a
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1a,CS1d
BC2,BC3,CS1c,BC4,CS1b,BC1,CS1d,CS1a
BC2,BC3,CS1c,BC4,CS1b,CS1d,BC1,CS1a
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1a,CS1b
BC2,BC3,CS1c,BC4,CS1d,BC1,CS1b,CS1a
BC2,BC3,CS1c,BC4,CS1d,CS1b,BC1,CS1a
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1a,CS1c
BC2,BC3,BC4,CS1d,CS1b,BC1,CS1c,CS1a
BC2,BC3,BC4,CS1d,CS1b,CS1c,BC1,CS1a
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1a,CS1b
BC2,BC3,BC4,CS1d,CS1c,BC1,CS1b,CS1a
BC2,BC3,BC4,CS1d,CS1c,CS1b,BC1,CS1a
BC2,BC4,CS1d,CS1b,BC1,CS1a,BC3,CS1c
BC2,BC4,CS1d,CS1b,BC1,BC3,CS1a,CS1c
BC2,BC4,CS1d,CS1b,BC1,BC3,CS1c,CS1a

BC2,BC4,CS1d,CS1b,BC3,CS1c,BC1,CS1a
BC3,CS1c,BC4,CS1d,BC1,CS1a,BC2,CS1b
BC3,CS1c,BC4,CS1d,BC1,BC2,CS1a,CS1b
BC3,CS1c,BC4,CS1d,BC1,BC2,CS1b,CS1a
BC3,CS1c,BC4,CS1d,BC2,CS1b,BC1,CS1a
BC3,BC4,CS1d,CS1c,BC1,CS1a,BC2,CS1b
BC3,BC4,CS1d,CS1c,BC1,BC2,CS1a,CS1b
BC3,BC4,CS1d,CS1c,BC1,BC2,CS1b,CS1a
BC3,BC4,CS1d,CS1c,BC2,CS1b,BC1,CS1a

Appendix K

All legal schedules for Case 6 without the time aspect of the program taken into account.

BC1,CS2a,BC2,CS1d,BC3,CS1b,BC4,CS1c
BC1,CS2a,BC2,CS1d,BC3,BC4,CS1b,CS1c
BC1,CS2a,BC2,CS1d,BC3,BC4,CS1c,CS1b
BC1,CS2a,BC2,CS1d,BC4,BC3,CS1b,CS1c
BC1,CS2a,BC2,CS1d,BC4,BC3,CS1c,CS1b
BC1,CS2a,BC2,CS1d,BC4,CS1c,BC3,CS1b
BC1,CS2a,BC2,BC3,CS1d,CS1b,BC4,CS1c
BC1,CS2a,BC2,BC3,CS1d,BC4,CS1b,CS1c
BC1,CS2a,BC2,BC3,CS1d,BC4,CS1c,CS1b
BC1,CS2a,BC2,BC3,CS1b,CS1d,BC4,CS1c
BC1,CS2a,BC2,BC3,CS1b,BC4,CS1d,CS1c
BC1,CS2a,BC2,BC3,CS1b,BC4,CS1c,CS1d
BC1,CS2a,BC2,BC3,BC4,CS1d,CS1b,CS1c
BC1,CS2a,BC2,BC3,BC4,CS1d,CS1c,CS1b
BC1,CS2a,BC2,BC3,BC4,CS1b,CS1d,CS1c
BC1,CS2a,BC2,BC3,BC4,CS1b,CS1c,CS1d
BC1,CS2a,BC2,BC3,BC4,CS1c,CS1d,CS1b
BC1,CS2a,BC2,BC3,BC4,CS1c,CS1b,CS1d
BC1,CS2a,BC2,BC4,CS1d,BC3,CS1b,CS1c
BC1,CS2a,BC2,BC4,CS1d,BC3,CS1c,CS1b
BC1,CS2a,BC2,BC4,CS1d,CS1c,BC3,CS1b
BC1,CS2a,BC2,BC4,BC3,CS1d,CS1b,CS1c
BC1,CS2a,BC2,BC4,BC3,CS1d,CS1c,CS1b
BC1,CS2a,BC2,BC4,BC3,CS1b,CS1d,CS1c
BC1,CS2a,BC2,BC4,BC3,CS1b,CS1c,CS1d
BC1,CS2a,BC2,BC4,BC3,CS1c,CS1d,CS1b
BC1,CS2a,BC2,BC4,BC3,CS1c,CS1b,CS1d
BC1,CS2a,BC2,BC4,CS1c,CS1d,BC3,CS1b
BC1,CS2a,BC2,BC4,CS1c,BC3,CS1d,CS1b
BC1,CS2a,BC2,BC4,CS1c,BC3,CS1b,CS1d
BC1,CS2a,BC3,BC2,CS1d,CS1b,BC4,CS1c
BC1,CS2a,BC3,BC2,CS1d,BC4,CS1b,CS1c
BC1,CS2a,BC3,BC2,CS1d,BC4,CS1c,CS1b
BC1,CS2a,BC3,BC2,CS1b,CS1d,BC4,CS1c
BC1,CS2a,BC3,BC2,CS1b,BC4,CS1d,CS1c
BC1,CS2a,BC3,BC2,CS1b,BC4,CS1c,CS1d
BC1,CS2a,BC3,BC2,BC4,CS1d,CS1b,CS1c
BC1,CS2a,BC3,BC2,BC4,CS1d,CS1c,CS1b
BC1,CS2a,BC3,BC2,BC4,CS1b,CS1d,CS1c
BC1,CS2a,BC3,BC2,BC4,CS1b,CS1c,CS1d
BC1,CS2a,BC3,BC2,BC4,CS1c,CS1d,CS1b

BC1,CS2a,BC3,BC2,BC4,CS1c,CS1b,CS1d
BC1,CS2a,BC3,CS1b,BC2,CS1d,BC4,CS1c
BC1,CS2a,BC3,CS1b,BC2,BC4,CS1d,CS1c
BC1,CS2a,BC3,CS1b,BC2,BC4,CS1c,CS1d
BC1,CS2a,BC3,CS1b,BC4,BC2,CS1d,CS1c
BC1,CS2a,BC3,CS1b,BC4,BC2,CS1c,CS1d
BC1,CS2a,BC3,CS1b,BC4,CS1c,BC2,CS1d
BC1,CS2a,BC3,BC4,BC2,CS1d,CS1b,CS1c
BC1,CS2a,BC3,BC4,BC2,CS1d,CS1c,CS1b
BC1,CS2a,BC3,BC4,BC2,CS1b,CS1d,CS1c
BC1,CS2a,BC3,BC4,BC2,CS1b,CS1c,CS1d
BC1,CS2a,BC3,BC4,BC2,CS1c,CS1d,CS1b
BC1,CS2a,BC3,BC4,BC2,CS1c,CS1b,CS1d
BC1,CS2a,BC3,BC4,CS1b,BC2,CS1d,CS1c
BC1,CS2a,BC3,BC4,CS1b,BC2,CS1c,CS1d
BC1,CS2a,BC3,BC4,CS1b,CS1c,BC2,CS1d
BC1,CS2a,BC3,BC4,CS1c,BC2,CS1d,CS1b
BC1,CS2a,BC3,BC4,CS1c,BC2,CS1b,CS1d
BC1,CS2a,BC3,BC4,CS1c,CS1b,BC2,CS1d
BC1,CS2a,BC4,BC2,CS1d,BC3,CS1b,CS1c
BC1,CS2a,BC4,BC2,CS1d,BC3,CS1c,CS1b
BC1,CS2a,BC4,BC2,CS1d,CS1c,BC3,CS1b
BC1,CS2a,BC4,BC2,BC3,CS1d,CS1b,CS1c
BC1,CS2a,BC4,BC2,BC3,CS1d,CS1c,CS1b
BC1,CS2a,BC4,BC2,BC3,CS1b,CS1d,CS1c
BC1,CS2a,BC4,BC2,BC3,CS1b,CS1c,CS1d
BC1,CS2a,BC4,BC2,BC3,CS1c,CS1d,CS1b
BC1,CS2a,BC4,BC2,BC3,CS1c,CS1b,CS1d
BC1,CS2a,BC4,BC2,CS1c,CS1d,BC3,CS1b
BC1,CS2a,BC4,BC2,CS1c,BC3,CS1d,CS1b
BC1,CS2a,BC4,BC2,CS1c,BC3,CS1b,CS1d
BC1,CS2a,BC4,BC3,BC2,CS1d,CS1b,CS1c
BC1,CS2a,BC4,BC3,BC2,CS1d,CS1c,CS1b
BC1,CS2a,BC4,BC3,BC2,CS1b,CS1d,CS1c
BC1,CS2a,BC4,BC3,BC2,CS1b,CS1c,CS1d
BC1,CS2a,BC4,BC3,BC2,CS1c,CS1d,CS1b
BC1,CS2a,BC4,BC3,BC2,CS1c,CS1b,CS1d
BC1,CS2a,BC4,BC3,CS1b,BC2,CS1d,CS1c
BC1,CS2a,BC4,BC3,CS1b,BC2,CS1c,CS1d
BC1,CS2a,BC4,BC3,CS1b,CS1c,BC2,CS1d
BC1,CS2a,BC4,BC3,CS1c,BC2,CS1d,CS1b
BC1,CS2a,BC4,BC3,CS1c,BC2,CS1b,CS1d
BC1,CS2a,BC4,BC3,CS1c,CS1b,BC2,CS1d
BC1,CS2a,BC4,CS1c,BC2,CS1d,BC3,CS1b
BC1,CS2a,BC4,CS1c,BC2,BC3,CS1d,CS1b
BC1,CS2a,BC4,CS1c,BC2,BC3,CS1b,CS1d

BC1,CS2a,BC4,CS1c,BC3,BC2,CS1d,CS1b
BC1,CS2a,BC4,CS1c,BC3,BC2,CS1b,CS1d
BC1,CS2a,BC4,CS1c,BC3,CS1b,BC2,CS1d
BC1,BC3,CS2a,BC2,CS1d,CS1b,BC4,CS1c
BC1,BC3,CS2a,BC2,CS1d,BC4,CS1b,CS1c
BC1,BC3,CS2a,BC2,CS1d,BC4,CS1c,CS1b
BC1,BC3,CS2a,BC2,CS1b,CS1d,BC4,CS1c
BC1,BC3,CS2a,BC2,CS1b,BC4,CS1d,CS1c
BC1,BC3,CS2a,BC2,CS1b,BC4,CS1c,CS1d
BC1,BC3,CS2a,BC2,BC4,CS1d,CS1b,CS1c
BC1,BC3,CS2a,BC2,BC4,CS1d,CS1c,CS1b
BC1,BC3,CS2a,BC2,BC4,CS1b,CS1d,CS1c
BC1,BC3,CS2a,BC2,BC4,CS1b,CS1c,CS1d
BC1,BC3,CS2a,BC2,BC4,CS1c,CS1d,CS1b
BC1,BC3,CS2a,BC2,BC4,CS1c,CS1b,CS1d
BC1,BC3,CS2a,CS1b,BC2,CS1d,BC4,CS1c
BC1,BC3,CS2a,CS1b,BC2,BC4,CS1d,CS1c
BC1,BC3,CS2a,CS1b,BC2,BC4,CS1c,CS1d
BC1,BC3,CS2a,CS1b,BC4,BC2,CS1d,CS1c
BC1,BC3,CS2a,CS1b,BC4,BC2,CS1c,CS1d
BC1,BC3,CS2a,CS1b,BC4,CS1c,BC2,CS1d
BC1,BC3,CS2a,BC4,BC2,CS1d,CS1b,CS1c
BC1,BC3,CS2a,BC4,BC2,CS1d,CS1c,CS1b
BC1,BC3,CS2a,BC4,BC2,CS1b,CS1d,CS1c
BC1,BC3,CS2a,BC4,BC2,CS1b,CS1c,CS1d
BC1,BC3,CS2a,BC4,BC2,CS1c,CS1d,CS1b
BC1,BC3,CS2a,BC4,BC2,CS1c,CS1b,CS1d
BC1,BC3,CS2a,BC4,CS1b,BC2,CS1d,CS1c
BC1,BC3,CS2a,BC4,CS1b,BC2,CS1c,CS1d
BC1,BC3,CS2a,BC4,CS1b,CS1c,BC2,CS1d
BC1,BC3,CS2a,BC4,CS1c,BC2,CS1d,CS1b
BC1,BC3,CS2a,BC4,CS1c,BC2,CS1b,CS1d
BC1,BC3,CS2a,BC4,CS1c,CS1b,BC2,CS1d
BC1,BC3,CS1b,CS2a,BC2,CS1d,BC4,CS1c
BC1,BC3,CS1b,CS2a,BC2,BC4,CS1d,CS1c
BC1,BC3,CS1b,CS2a,BC2,BC4,CS1c,CS1d
BC1,BC3,CS1b,CS2a,BC4,BC2,CS1d,CS1c
BC1,BC3,CS1b,CS2a,BC4,BC2,CS1c,CS1d
BC1,BC3,CS1b,CS2a,BC4,CS1c,BC2,CS1d
BC1,BC3,CS1b,BC4,CS2a,BC2,CS1d,CS1c
BC1,BC3,CS1b,BC4,CS2a,BC2,CS1c,CS1d
BC1,BC3,CS1b,BC4,CS2a,CS1c,BC2,CS1d
BC1,BC3,CS1b,BC4,CS1c,CS2a,BC2,CS1d
BC1,BC3,BC4,CS2a,BC2,CS1d,CS1b,CS1c
BC1,BC3,BC4,CS2a,BC2,CS1d,CS1c,CS1b
BC1,BC3,BC4,CS2a,BC2,CS1b,CS1d,CS1c

BC1,BC3,BC4,CS2a,BC2,CS1b,CS1c,CS1d
BC1,BC3,BC4,CS2a,BC2,CS1c,CS1d,CS1b
BC1,BC3,BC4,CS2a,BC2,CS1c,CS1b,CS1d
BC1,BC3,BC4,CS2a,CS1b,BC2,CS1d,CS1c
BC1,BC3,BC4,CS2a,CS1b,BC2,CS1c,CS1d
BC1,BC3,BC4,CS2a,CS1b,CS1c,BC2,CS1d
BC1,BC3,BC4,CS2a,CS1c,BC2,CS1d,CS1b
BC1,BC3,BC4,CS2a,CS1c,BC2,CS1b,CS1d
BC1,BC3,BC4,CS2a,CS1c,CS1b,BC2,CS1d
BC1,BC3,BC4,CS1b,CS2a,BC2,CS1d,CS1c
BC1,BC3,BC4,CS1b,CS2a,BC2,CS1c,CS1d
BC1,BC3,BC4,CS1b,CS2a,CS1c,BC2,CS1d
BC1,BC3,BC4,CS1b,CS1c,CS2a,BC2,CS1d
BC1,BC3,BC4,CS1c,CS2a,BC2,CS1d,CS1b
BC1,BC3,BC4,CS1c,CS2a,BC2,CS1b,CS1d
BC1,BC3,BC4,CS1c,CS2a,CS1b,BC2,CS1d
BC1,BC3,BC4,CS1c,CS1b,CS2a,BC2,CS1d
BC1,BC4,CS2a,BC2,CS1d,BC3,CS1b,CS1c
BC1,BC4,CS2a,BC2,CS1d,BC3,CS1c,CS1b
BC1,BC4,CS2a,BC2,CS1d,CS1c,BC3,CS1b
BC1,BC4,CS2a,BC2,BC3,CS1d,CS1b,CS1c
BC1,BC4,CS2a,BC2,BC3,CS1d,CS1c,CS1b
BC1,BC4,CS2a,BC2,BC3,CS1b,CS1d,CS1c
BC1,BC4,CS2a,BC2,BC3,CS1b,CS1c,CS1d
BC1,BC4,CS2a,BC2,BC3,CS1c,CS1d,CS1b
BC1,BC4,CS2a,BC2,BC3,CS1c,CS1b,CS1d
BC1,BC4,CS2a,BC2,CS1c,CS1d,BC3,CS1b
BC1,BC4,CS2a,BC2,CS1c,BC3,CS1d,CS1b
BC1,BC4,CS2a,BC2,CS1c,BC3,CS1b,CS1d
BC1,BC4,CS2a,BC3,BC2,CS1d,CS1b,CS1c
BC1,BC4,CS2a,BC3,BC2,CS1d,CS1c,CS1b
BC1,BC4,CS2a,BC3,BC2,CS1b,CS1d,CS1c
BC1,BC4,CS2a,BC3,BC2,CS1b,CS1c,CS1d
BC1,BC4,CS2a,BC3,BC2,CS1c,CS1d,CS1b
BC1,BC4,CS2a,BC3,BC2,CS1c,CS1b,CS1d
BC1,BC4,CS2a,BC3,CS1b,BC2,CS1d,CS1c
BC1,BC4,CS2a,BC3,CS1b,BC2,CS1c,CS1d
BC1,BC4,CS2a,BC3,CS1b,CS1c,BC2,CS1d
BC1,BC4,CS2a,BC3,CS1c,BC2,CS1d,CS1b
BC1,BC4,CS2a,BC3,CS1c,BC2,CS1b,CS1d
BC1,BC4,CS2a,BC3,CS1c,CS1b,BC2,CS1d
BC1,BC4,CS2a,CS1c,BC2,CS1d,BC3,CS1b
BC1,BC4,CS2a,CS1c,BC2,BC3,CS1d,CS1b
BC1,BC4,CS2a,CS1c,BC2,BC3,CS1b,CS1d
BC1,BC4,CS2a,CS1c,BC3,BC2,CS1d,CS1b
BC1,BC4,CS2a,CS1c,BC3,BC2,CS1b,CS1d

BC1,BC4,CS2a,CS1c,BC3,CS1b,BC2,CS1d
BC1,BC4,BC3,CS2a,BC2,CS1d,CS1b,CS1c
BC1,BC4,BC3,CS2a,BC2,CS1d,CS1c,CS1b
BC1,BC4,BC3,CS2a,BC2,CS1b,CS1d,CS1c
BC1,BC4,BC3,CS2a,BC2,CS1b,CS1c,CS1d
BC1,BC4,BC3,CS2a,BC2,CS1c,CS1d,CS1b
BC1,BC4,BC3,CS2a,BC2,CS1c,CS1b,CS1d
BC1,BC4,BC3,CS2a,CS1b,BC2,CS1d,CS1c
BC1,BC4,BC3,CS2a,CS1b,BC2,CS1c,CS1d
BC1,BC4,BC3,CS2a,CS1b,CS1c,BC2,CS1d
BC1,BC4,BC3,CS2a,CS1c,BC2,CS1d,CS1b
BC1,BC4,BC3,CS2a,CS1c,BC2,CS1b,CS1d
BC1,BC4,BC3,CS2a,CS1c,CS1b,BC2,CS1d
BC1,BC4,BC3,CS1b,CS2a,BC2,CS1d,CS1c
BC1,BC4,BC3,CS1b,CS2a,BC2,CS1c,CS1d
BC1,BC4,BC3,CS1b,CS2a,CS1c,BC2,CS1d
BC1,BC4,BC3,CS1b,CS1c,CS2a,BC2,CS1d
BC1,BC4,BC3,CS1c,CS2a,BC2,CS1d,CS1b
BC1,BC4,BC3,CS1c,CS2a,BC2,CS1b,CS1d
BC1,BC4,BC3,CS1c,CS2a,CS1b,BC2,CS1d
BC1,BC4,BC3,CS1c,CS1b,CS2a,BC2,CS1d
BC1,BC4,CS1c,CS2a,BC2,CS1d,BC3,CS1b
BC1,BC4,CS1c,CS2a,BC2,BC3,CS1d,CS1b
BC1,BC4,CS1c,CS2a,BC2,BC3,CS1b,CS1d
BC1,BC4,CS1c,CS2a,BC3,BC2,CS1d,CS1b
BC1,BC4,CS1c,CS2a,BC3,BC2,CS1b,CS1d
BC1,BC4,CS1c,CS2a,BC3,CS1b,BC2,CS1d
BC1,BC4,CS1c,BC3,CS2a,BC2,CS1d,CS1b
BC1,BC4,CS1c,BC3,CS2a,BC2,CS1b,CS1d
BC1,BC4,CS1c,BC3,CS2a,CS1b,BC2,CS1d
BC1,BC4,CS1c,BC3,CS1b,CS2a,BC2,CS1d
BC3,BC1,CS2a,BC2,CS1d,CS1b,BC4,CS1c
BC3,BC1,CS2a,BC2,CS1d,BC4,CS1b,CS1c
BC3,BC1,CS2a,BC2,CS1d,BC4,CS1c,CS1b
BC3,BC1,CS2a,BC2,CS1b,CS1d,BC4,CS1c
BC3,BC1,CS2a,BC2,CS1b,BC4,CS1d,CS1c
BC3,BC1,CS2a,BC2,CS1b,BC4,CS1c,CS1d
BC3,BC1,CS2a,BC2,BC4,CS1d,CS1b,CS1c
BC3,BC1,CS2a,BC2,BC4,CS1d,CS1c,CS1b
BC3,BC1,CS2a,BC2,BC4,CS1b,CS1d,CS1c
BC3,BC1,CS2a,BC2,BC4,CS1b,CS1c,CS1d
BC3,BC1,CS2a,BC2,BC4,CS1c,CS1d,CS1b
BC3,BC1,CS2a,BC2,BC4,CS1c,CS1b,CS1d
BC3,BC1,CS2a,CS1b,BC2,CS1d,BC4,CS1c
BC3,BC1,CS2a,CS1b,BC2,BC4,CS1d,CS1c
BC3,BC1,CS2a,CS1b,BC2,BC4,CS1c,CS1d

BC3,BC1,CS2a,CS1b,BC4,BC2,CS1d,CS1c
BC3,BC1,CS2a,CS1b,BC4,BC2,CS1c,CS1d
BC3,BC1,CS2a,CS1b,BC4,CS1c,BC2,CS1d
BC3,BC1,CS2a,BC4,BC2,CS1d,CS1b,CS1c
BC3,BC1,CS2a,BC4,BC2,CS1d,CS1c,CS1b
BC3,BC1,CS2a,BC4,BC2,CS1b,CS1d,CS1c
BC3,BC1,CS2a,BC4,BC2,CS1b,CS1c,CS1d
BC3,BC1,CS2a,BC4,BC2,CS1c,CS1d,CS1b
BC3,BC1,CS2a,BC4,BC2,CS1c,CS1b,CS1d
BC3,BC1,CS2a,BC4,CS1b,BC2,CS1d,CS1c
BC3,BC1,CS2a,BC4,CS1b,BC2,CS1c,CS1d
BC3,BC1,CS2a,BC4,CS1b,CS1c,BC2,CS1d
BC3,BC1,CS2a,BC4,CS1c,BC2,CS1d,CS1b
BC3,BC1,CS2a,BC4,CS1c,BC2,CS1b,CS1d
BC3,BC1,CS2a,BC4,CS1c,CS1b,BC2,CS1d
BC3,BC1,CS1b,CS2a,BC2,CS1d,BC4,CS1c
BC3,BC1,CS1b,CS2a,BC2,BC4,CS1d,CS1c
BC3,BC1,CS1b,CS2a,BC2,BC4,CS1c,CS1d
BC3,BC1,CS1b,CS2a,BC4,BC2,CS1d,CS1c
BC3,BC1,CS1b,CS2a,BC4,BC2,CS1c,CS1d
BC3,BC1,CS1b,CS2a,BC4,CS1c,BC2,CS1d
BC3,BC1,CS1b,BC4,CS2a,BC2,CS1d,CS1c
BC3,BC1,CS1b,BC4,CS2a,BC2,CS1c,CS1d
BC3,BC1,CS1b,BC4,CS2a,CS1c,BC2,CS1d
BC3,BC1,CS1b,BC4,CS1c,CS2a,BC2,CS1d
BC3,BC1,BC4,CS2a,BC2,CS1d,CS1b,CS1c
BC3,BC1,BC4,CS2a,BC2,CS1d,CS1c,CS1b
BC3,BC1,BC4,CS2a,BC2,CS1b,CS1d,CS1c
BC3,BC1,BC4,CS2a,BC2,CS1b,CS1c,CS1d
BC3,BC1,BC4,CS2a,BC2,CS1c,CS1d,CS1b
BC3,BC1,BC4,CS2a,BC2,CS1c,CS1b,CS1d
BC3,BC1,BC4,CS2a,CS1b,BC2,CS1d,CS1c
BC3,BC1,BC4,CS2a,CS1b,BC2,CS1c,CS1d
BC3,BC1,BC4,CS2a,CS1b,CS1c,BC2,CS1d
BC3,BC1,BC4,CS2a,CS1c,BC2,CS1d,CS1b
BC3,BC1,BC4,CS2a,CS1c,BC2,CS1b,CS1d
BC3,BC1,BC4,CS2a,CS1c,CS1b,BC2,CS1d
BC3,BC1,BC4,CS1b,CS2a,BC2,CS1d,CS1c
BC3,BC1,BC4,CS1b,CS2a,BC2,CS1c,CS1d
BC3,BC1,BC4,CS1b,CS2a,CS1c,BC2,CS1d
BC3,BC1,BC4,CS1b,CS1c,CS2a,BC2,CS1d
BC3,BC1,BC4,CS1c,CS2a,BC2,CS1d,CS1b
BC3,BC1,BC4,CS1c,CS2a,BC2,CS1b,CS1d
BC3,BC1,BC4,CS1c,CS2a,CS1b,BC2,CS1d
BC3,BC1,BC4,CS1c,CS1b,CS2a,BC2,CS1d
BC3,CS1b,BC1,CS2a,BC2,CS1d,BC4,CS1c

BC3,CS1b,BC1,CS2a,BC2,BC4,CS1d,CS1c
BC3,CS1b,BC1,CS2a,BC2,BC4,CS1c,CS1d
BC3,CS1b,BC1,CS2a,BC4,BC2,CS1d,CS1c
BC3,CS1b,BC1,CS2a,BC4,BC2,CS1c,CS1d
BC3,CS1b,BC1,CS2a,BC4,CS1c,BC2,CS1d
BC3,CS1b,BC1,BC4,CS2a,BC2,CS1d,CS1c
BC3,CS1b,BC1,BC4,CS2a,BC2,CS1c,CS1d
BC3,CS1b,BC1,BC4,CS2a,CS1c,BC2,CS1d
BC3,CS1b,BC1,BC4,CS1c,CS2a,BC2,CS1d
BC3,CS1b,BC4,BC1,CS2a,BC2,CS1d,CS1c
BC3,CS1b,BC4,BC1,CS2a,BC2,CS1c,CS1d
BC3,CS1b,BC4,BC1,CS2a,CS1c,BC2,CS1d
BC3,CS1b,BC4,BC1,CS1c,CS2a,BC2,CS1d
BC3,CS1b,BC4,CS1c,BC1,CS2a,BC2,CS1d
BC3,BC4,BC1,CS2a,BC2,CS1d,CS1b,CS1c
BC3,BC4,BC1,CS2a,BC2,CS1d,CS1c,CS1b
BC3,BC4,BC1,CS2a,BC2,CS1b,CS1d,CS1c
BC3,BC4,BC1,CS2a,BC2,CS1b,CS1c,CS1d
BC3,BC4,BC1,CS2a,BC2,CS1c,CS1d,CS1b
BC3,BC4,BC1,CS2a,BC2,CS1c,CS1b,CS1d
BC3,BC4,BC1,CS2a,CS1b,BC2,CS1d,CS1c
BC3,BC4,BC1,CS2a,CS1b,BC2,CS1c,CS1d
BC3,BC4,BC1,CS2a,CS1b,CS1c,BC2,CS1d
BC3,BC4,BC1,CS2a,CS1c,BC2,CS1d,CS1b
BC3,BC4,BC1,CS2a,CS1c,BC2,CS1b,CS1d
BC3,BC4,BC1,CS2a,CS1c,CS1b,BC2,CS1d
BC3,BC4,BC1,CS1b,CS2a,BC2,CS1d,CS1c
BC3,BC4,BC1,CS1b,CS2a,BC2,CS1c,CS1d
BC3,BC4,BC1,CS1b,CS2a,CS1c,BC2,CS1d
BC3,BC4,BC1,CS1b,CS1c,CS2a,BC2,CS1d
BC3,BC4,BC1,CS1c,CS2a,BC2,CS1d,CS1b
BC3,BC4,BC1,CS1c,CS2a,BC2,CS1b,CS1d
BC3,BC4,BC1,CS1c,CS2a,CS1b,BC2,CS1d
BC3,BC4,BC1,CS1c,CS1b,CS2a,BC2,CS1d
BC3,BC4,CS1b,BC1,CS2a,BC2,CS1d,CS1c
BC3,BC4,CS1b,BC1,CS2a,BC2,CS1c,CS1d
BC3,BC4,CS1b,BC1,CS2a,CS1c,BC2,CS1d
BC3,BC4,CS1b,BC1,CS1c,CS2a,BC2,CS1d
BC3,BC4,CS1b,CS1c,BC1,CS2a,BC2,CS1d
BC3,BC4,CS1c,BC1,CS2a,BC2,CS1d,CS1b
BC3,BC4,CS1c,BC1,CS2a,BC2,CS1b,CS1d
BC3,BC4,CS1c,BC1,CS2a,CS1b,BC2,CS1d
BC3,BC4,CS1c,BC1,CS1b,CS2a,BC2,CS1d
BC3,BC4,CS1c,CS1b,BC1,CS2a,BC2,CS1d
BC4,BC1,CS2a,BC2,CS1d,BC3,CS1b,CS1c
BC4,BC1,CS2a,BC2,CS1d,BC3,CS1c,CS1b

BC4,BC1,CS2a,BC2,CS1d,CS1c,BC3,CS1b
BC4,BC1,CS2a,BC2,BC3,CS1d,CS1b,CS1c
BC4,BC1,CS2a,BC2,BC3,CS1d,CS1c,CS1b
BC4,BC1,CS2a,BC2,BC3,CS1b,CS1d,CS1c
BC4,BC1,CS2a,BC2,BC3,CS1b,CS1c,CS1d
BC4,BC1,CS2a,BC2,BC3,CS1c,CS1d,CS1b
BC4,BC1,CS2a,BC2,BC3,CS1c,CS1b,CS1d
BC4,BC1,CS2a,BC2,CS1c,CS1d,BC3,CS1b
BC4,BC1,CS2a,BC2,CS1c,BC3,CS1d,CS1b
BC4,BC1,CS2a,BC2,CS1c,BC3,CS1b,CS1d
BC4,BC1,CS2a,BC3,BC2,CS1d,CS1b,CS1c
BC4,BC1,CS2a,BC3,BC2,CS1d,CS1c,CS1b
BC4,BC1,CS2a,BC3,BC2,CS1b,CS1d,CS1c
BC4,BC1,CS2a,BC3,BC2,CS1b,CS1c,CS1d
BC4,BC1,CS2a,BC3,BC2,CS1c,CS1d,CS1b
BC4,BC1,CS2a,BC3,BC2,CS1c,CS1b,CS1d
BC4,BC1,CS2a,BC3,CS1b,BC2,CS1d,CS1c
BC4,BC1,CS2a,BC3,CS1b,BC2,CS1c,CS1d
BC4,BC1,CS2a,BC3,CS1b,CS1c,BC2,CS1d
BC4,BC1,CS2a,BC3,CS1c,BC2,CS1d,CS1b
BC4,BC1,CS2a,BC3,CS1c,BC2,CS1b,CS1d
BC4,BC1,CS2a,BC3,CS1c,CS1b,BC2,CS1d
BC4,BC1,CS2a,CS1c,BC2,CS1d,BC3,CS1b
BC4,BC1,CS2a,CS1c,BC2,BC3,CS1d,CS1b
BC4,BC1,CS2a,CS1c,BC2,BC3,CS1b,CS1d
BC4,BC1,CS2a,CS1c,BC3,BC2,CS1d,CS1b
BC4,BC1,CS2a,CS1c,BC3,BC2,CS1b,CS1d
BC4,BC1,CS2a,CS1c,BC3,CS1b,BC2,CS1d
BC4,BC1,BC3,CS2a,BC2,CS1d,CS1b,CS1c
BC4,BC1,BC3,CS2a,BC2,CS1d,CS1c,CS1b
BC4,BC1,BC3,CS2a,BC2,CS1b,CS1d,CS1c
BC4,BC1,BC3,CS2a,BC2,CS1b,CS1c,CS1d
BC4,BC1,BC3,CS2a,BC2,CS1c,CS1d,CS1b
BC4,BC1,BC3,CS2a,BC2,CS1c,CS1b,CS1d
BC4,BC1,BC3,CS2a,CS1b,BC2,CS1d,CS1c
BC4,BC1,BC3,CS2a,CS1b,BC2,CS1c,CS1d
BC4,BC1,BC3,CS2a,CS1b,CS1c,BC2,CS1d
BC4,BC1,BC3,CS2a,CS1c,BC2,CS1d,CS1b
BC4,BC1,BC3,CS2a,CS1c,BC2,CS1b,CS1d
BC4,BC1,BC3,CS2a,CS1c,CS1b,BC2,CS1d
BC4,BC1,BC3,CS1b,CS2a,BC2,CS1d,CS1c
BC4,BC1,BC3,CS1b,CS2a,BC2,CS1c,CS1d
BC4,BC1,BC3,CS1b,CS2a,CS1c,BC2,CS1d
BC4,BC1,BC3,CS1b,CS1c,CS2a,BC2,CS1d
BC4,BC1,BC3,CS1c,CS2a,BC2,CS1d,CS1b
BC4,BC1,BC3,CS1c,CS2a,BC2,CS1b,CS1d

BC4,BC1,BC3,CS1c,CS2a,CS1b,BC2,CS1d
BC4,BC1,BC3,CS1c,CS1b,CS2a,BC2,CS1d
BC4,BC1,CS1c,CS2a,BC2,CS1d,BC3,CS1b
BC4,BC1,CS1c,CS2a,BC2,BC3,CS1d,CS1b
BC4,BC1,CS1c,CS2a,BC2,BC3,CS1b,CS1d
BC4,BC1,CS1c,CS2a,BC3,BC2,CS1d,CS1b
BC4,BC1,CS1c,CS2a,BC3,BC2,CS1b,CS1d
BC4,BC1,CS1c,CS2a,BC3,CS1b,BC2,CS1d
BC4,BC1,CS1c,BC3,CS2a,BC2,CS1d,CS1b
BC4,BC1,CS1c,BC3,CS2a,BC2,CS1b,CS1d
BC4,BC1,CS1c,BC3,CS2a,CS1b,BC2,CS1d
BC4,BC1,CS1c,BC3,CS1b,CS2a,BC2,CS1d
BC4,BC3,BC1,CS2a,BC2,CS1d,CS1b,CS1c
BC4,BC3,BC1,CS2a,BC2,CS1d,CS1c,CS1b
BC4,BC3,BC1,CS2a,BC2,CS1b,CS1d,CS1c
BC4,BC3,BC1,CS2a,BC2,CS1b,CS1c,CS1d
BC4,BC3,BC1,CS2a,BC2,CS1c,CS1d,CS1b
BC4,BC3,BC1,CS2a,BC2,CS1c,CS1b,CS1d
BC4,BC3,BC1,CS2a,CS1b,BC2,CS1d,CS1c
BC4,BC3,BC1,CS2a,CS1b,BC2,CS1c,CS1d
BC4,BC3,BC1,CS2a,CS1b,CS1c,BC2,CS1d
BC4,BC3,BC1,CS2a,CS1c,BC2,CS1d,CS1b
BC4,BC3,BC1,CS2a,CS1c,BC2,CS1b,CS1d
BC4,BC3,BC1,CS2a,CS1c,CS1b,BC2,CS1d
BC4,BC3,BC1,CS1b,CS2a,BC2,CS1d,CS1c
BC4,BC3,BC1,CS1b,CS2a,BC2,CS1c,CS1d
BC4,BC3,BC1,CS1b,CS2a,CS1c,BC2,CS1d
BC4,BC3,BC1,CS1b,CS1c,CS2a,BC2,CS1d
BC4,BC3,BC1,CS1c,CS2a,BC2,CS1d,CS1b
BC4,BC3,BC1,CS1c,CS2a,BC2,CS1b,CS1d
BC4,BC3,BC1,CS1c,CS2a,CS1b,BC2,CS1d
BC4,BC3,BC1,CS1c,CS1b,CS2a,BC2,CS1d
BC4,BC3,CS1b,BC1,CS2a,BC2,CS1d,CS1c
BC4,BC3,CS1b,BC1,CS2a,BC2,CS1c,CS1d
BC4,BC3,CS1b,BC1,CS2a,CS1c,BC2,CS1d
BC4,BC3,CS1b,BC1,CS1c,CS2a,BC2,CS1d
BC4,BC3,CS1b,CS1c,BC1,CS2a,BC2,CS1d
BC4,BC3,CS1c,BC1,CS2a,BC2,CS1d,CS1b
BC4,BC3,CS1c,BC1,CS2a,BC2,CS1b,CS1d
BC4,BC3,CS1c,BC1,CS2a,CS1b,BC2,CS1d
BC4,BC3,CS1c,BC1,CS1b,CS2a,BC2,CS1d
BC4,BC3,CS1c,CS1b,BC1,CS2a,BC2,CS1d
BC4,CS1c,BC1,CS2a,BC2,CS1d,BC3,CS1b
BC4,CS1c,BC1,CS2a,BC2,BC3,CS1d,CS1b
BC4,CS1c,BC1,CS2a,BC2,BC3,CS1b,CS1d
BC4,CS1c,BC1,CS2a,BC3,BC2,CS1d,CS1b

BC4,CS1c,BC1,CS2a,BC3,BC2,CS1b,CS1d
BC4,CS1c,BC1,CS2a,BC3,CS1b,BC2,CS1d
BC4,CS1c,BC1,BC3,CS2a,BC2,CS1d,CS1b
BC4,CS1c,BC1,BC3,CS2a,BC2,CS1b,CS1d
BC4,CS1c,BC1,BC3,CS2a,CS1b,BC2,CS1d
BC4,CS1c,BC1,BC3,CS1b,CS2a,BC2,CS1d
BC4,CS1c,BC3,BC1,CS2a,BC2,CS1d,CS1b
BC4,CS1c,BC3,BC1,CS2a,BC2,CS1b,CS1d
BC4,CS1c,BC3,BC1,CS2a,CS1b,BC2,CS1d
BC4,CS1c,BC3,BC1,CS1b,CS2a,BC2,CS1d
BC4,CS1c,BC3,CS1b,BC1,CS2a,BC2,CS1d

Appendix L

All legal schedules for Case 6 taking the time aspect of the program into account.

BC1,CS2a,BC2,BC3,CS1b,BC4,CS1c,CS1d
BC1,CS2a,BC2,BC3,BC4,CS1b,CS1c,CS1d
BC1,CS2a,BC2,BC3,BC4,CS1c,CS1b,CS1d
BC1,CS2a,BC2,BC4,BC3,CS1b,CS1c,CS1d
BC1,CS2a,BC2,BC4,BC3,CS1c,CS1b,CS1d
BC1,CS2a,BC2,BC4,CS1c,BC3,CS1b,CS1d