



HAL
open science

Word-Size RMR Trade-offs for Recoverable Mutual Exclusion

David Yu Cheng Chan, George Giakkoupis, Philipp Woelfel

► **To cite this version:**

David Yu Cheng Chan, George Giakkoupis, Philipp Woelfel. Word-Size RMR Trade-offs for Recoverable Mutual Exclusion. 2023. hal-04098408

HAL Id: hal-04098408

<https://inria.hal.science/hal-04098408v1>

Preprint submitted on 16 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Word-Size RMR Trade-offs for Recoverable Mutual Exclusion

DAVID YU CHENG CHAN, University of Calgary, Canada

GEORGE GIAKKOUPIS, Inria, Univ Rennes, CNRS, IRISA, France

PHILIPP WOELFEL, University of Calgary, Canada

We present tradeoffs between RMR complexity and memory word size for recoverable mutual exclusion (RME) algorithms using arbitrary synchronization primitives. Assuming that each memory location stores w bits, we show that n -process mutual exclusion has an RMR complexity of at least $\Omega(\min\{\log_w n, \log n/\log \log n\})$ on the DSM and the CC model. For $w = (\log n)^{\Omega(1)}$, our lower bound asymptotically matches an upper bound by Katzan and Morrison [18], whose RME mutual exclusion algorithm employs w -bit fetch-and-add operations. Our lower bound is the first one that does not restrict the type of atomic operations that can be executed on a memory location.

CCS Concepts: • **Theory of computation** → **Shared memory algorithms**; • **Software and its engineering** → **Mutual exclusion**.

Additional Key Words and Phrases: mutual exclusion, recoverable mutual exclusion, critical section, RME, concurrency, shared memory, fault tolerance

1 INTRODUCTION

Mutual exclusion [7] is arguably the most important tool for synchronization in concurrent algorithms. A mutual exclusion algorithm is used to serialize access to a piece of code, called the *critical section*. To get access to the critical section, a process needs to complete a protocol called *entry section*. Once it is finished with the critical section, it indicates so by executing the *exit section*. The *mutual exclusion* property guarantees that no two processes are in the critical section at the same time. In addition, it is generally required that these algorithms are *deadlock-free*, meaning that as long as all participating processes keep taking steps, not all of them can be stuck in the entry or exit section.

The conventional model assumes a completely fault-free system, in which processes cannot crash. Motivated by recent advances in non-volatile memory technology, Golab and Ramaraju [11] defined the *recoverable mutual exclusion* (RME) problem. Here, process crashes are allowed, but shared memory is persisted. Any individual process can crash at any point in time, upon which its entire local state is reset, including any local variables, while the state of the shared memory is unaffected. After crashing, a process starts a recovery procedure and then resumes its mutual exclusion protocol. Since its introduction in 2015, the RME problem has been studied thoroughly [3, 4, 6, 9–12, 14–16, 18, 21].

The number of steps a process needs to execute in the entry section of a mutual exclusion algorithm cannot be bounded, because processes may have to busy-wait until the critical section becomes free. To analytically predict the performance of (conventional and recoverable) mutual exclusion algorithms, it is standard to analyze the maximum number of *remote memory references* (RMRs) that can be incurred during the entry and exit protocols. RMRs capture shared memory operations that require expensive communication through the processor-memory interconnect, or between processors. In the *distributed shared memory* (DSM) model, memory is partitioned into segments, each belonging to a different process. Whenever a process accesses shared memory outside its own segment, an RMR is incurred. In the *cache-coherent* (CC) model, each shared memory access incurs an RMR, except for a read operation by a process holding a valid cache copy of the accessed memory location.

In general, the RMR complexity of shared memory problems is very sensitive to the types of atomic operations supported by the system. Often, the minimal assumption is that atomic read

and write operations are supported. More powerful operations fetch (and return) the value of a memory location, but at the same time modify the value stored at that location. For example, fetch-and-increment increments it, fetch-and-add adds an integer value (given as a parameter) to it, fetch-and-store writes a new value, and compare-and-swap conditionally writes a value, provided that the current value matches another given parameter. All these operations are supported by many of today’s hardware architectures.

Much is known about the RMR complexity of conventional mutual exclusion for n processes. For systems that support only atomic read and write operations, there are mutual exclusion algorithms with RMR complexity $O(\log n)$ [22], which is optimal [2]. These bounds remain the same, even if the system supports compare-and-swap, which can be implemented from registers with constant RMR complexity [8]. By employing fetch-and-store or fetch-and-increment, the RMR complexity can be reduced to $O(1)$ [5, 19, 20].

The dependency of the RMR complexity of mutual exclusion on the word size w (measured in bits) of memory locations has not been studied explicitly, because existing algorithms and lower bounds are independent of w , as long as $w = \Omega(\log n)$. It is common to assume $w = \Omega(\log n)$, so that a process can at least store its own ID using only a constant number of memory words.

For recoverable mutual exclusion on systems supporting atomic fetch-and-store operations, the best known algorithms have an RMR complexity of $\Theta(\log n / \log \log n)$ [9, 14]. (Here, the RMR complexity is the maximum number of RMRs a process can incur in any *passage*, which begins with the process’s entry section or recover protocol, and ends when the process crashes or finishes its subsequent exit section.) Recently, Chan and Woelfel [4] showed that this is optimal for algorithms using only read, fetch-and-store, fetch-and-increment, and compare-and-swap operations. For the upper bounds, a word size w of $O(\log n)$ bits is sufficient, but the matching lower bound is true for any word size.

Better RMR complexity has been achieved for RME by using artificially defined operations that can atomically change two memory locations, and where the value written to one location depends on the value of the other location [9, 12]. Such operations do not exist in real systems, and have only been defined to study what it takes to improve the RMR complexity of RME. Throughout this paper we will make the standard assumption that each operation can only affect a single memory location.

Katzan and Morrison [18] observed that using fetch-and-add operations, which are supported by common hardware, the RMR complexity can be reduced to $O(1)$, assuming a word size of $n^{\Omega(1)}$ bits. More generally, their algorithm has an RMR complexity of $O(\log_w n)$ for w -bit fetch-and-add objects. One can argue that it is unrealistic to assume that the size of memory locations is polynomial in the number n of processors.

These results lead to two fundamental questions: First, are there atomic shared memory operations that allow us to solve RME with an RMR complexity of $o(\log n / \log \log n)$, under the common assumption that the word size is logarithmic, or at least poly-logarithmic in n ? Second, is it possible to improve upon the trade-off between word-size and RMR complexity that the algorithm by Katzan and Morrison exhibits? In this paper, we provide negative answers to both questions:

THEOREM 1. *Any deadlock-free n -process RME algorithm tolerating individual process crashes, where all shared memory locations store values from a domain of size 2^w (but support arbitrary atomic operations), has RMR complexity $\Omega\left(\min\left(\log_w n, \frac{\log n}{\log \log n}\right)\right)$ in the CC and the DSM model.*

The lower bound is asymptotically tight for $w \geq (\log n)^\epsilon$, for any $\epsilon > 0$, as it matches the upper bound of [18]. It shows that it is not possible to improve the algorithms with $O(\log n / \log \log n)$ RMR complexity [9, 14] on systems whose memory locations can store only poly-logarithmic many

bits. As far as we know, all known RME algorithms implicitly assume $w = \Omega(\log n)$, and RMR upper bounds for smaller word sizes have not been studied.

This is the first RMR lower bound that does not rely on restricting the types of shared memory operations an algorithm can use (but instead restricts the word size). It is also the first trade-off between word size and RMR complexity. In fact, as far as we know, no such lower bounds or trade-offs have previously been observed for *any* shared memory problem.

1.1 Technical Contribution

Our proof builds on ideas of Chan and Woelfel [4], which in turn uses the framework of Anderson and Kim [1], but requires some deeper combinatorial insights and has to deal with several technical difficulties in a novel way. In this section we sketch the core ideas of [1, 4], and explain how our new approach is different.

Anderson and Kim’s lower bound of $\Omega(\log n / \log \log n)$ applies to conventional (non-recoverable) *one-shot* mutual exclusion, i.e., each process tries to enter and exit the critical section once, and then stops taking steps. This lower bound assumes that each process can only perform read and write operations.

The idea is to construct a sequence of executions, E_0, E_1, \dots , where E_i comprises i rounds. With each execution E_i we associate a set of *active* processes that perform exactly one RMR in each of the rounds $1, \dots, i$ and at the end of E_i are poised to perform another RMR. Moreover, in each round some processes may be forced to run to completion upon which they are *finished*, and others are removed from the execution (i.e., we determine a new execution, where the removed processes do not take any steps). The remaining processes continue to be active. The main invariant is that any active process in round i can be removed from E_i without affecting any of the non-removed processes participating in E_i , and without making any non-removed process “visible” on a register. For example, if in round 1 processes p_1, p_2 , and p_3 write to a register R in this order, then p_3 cannot be active (and thus must be finished) at the end of the round, as removing p_3 would make p_2 visible on R . On the other hand, p_1 and p_2 can both be active, as neither has seen (or discovered) the other. If in some round a process q_1 writes a register and then immediately after that a process q_2 reads it, then it is not possible that q_1 and q_2 both remain active, because q_2 has discovered q_1 , and so removing q_1 from the execution would affect q_2 .

The lower bound then proceeds by constructing E_0, E_1, \dots, E_ℓ iteratively for some $\ell \in \Omega(\log n / \log \log n)$, such that E_ℓ has still at least 2 active processes (which must have performed ℓ RMRs).

To construct E_i from E_{i-1} , one distinguishes between low and high contention rounds. A low contention round occurs if at the beginning of E_i the majority of processes are poised to access a “low contention” register, which means fewer than k processes are about to access it, where k is a poly-logarithmic threshold. Otherwise, the round is called a high contention round.

We will focus on the high contention round, which is the one where most of our new ideas are applied. For the purpose of simplicity, assume that on each register there are exactly k processes poised to perform their next shared memory step, which is a write that will incur an RMR. Then we can schedule the writes in arbitrary order and after that let the last writer, p , run to completion, removing any active processes that p may otherwise discover. (Recall that by the invariants of this construction, we can always remove arbitrary active processes without affecting any other processes.) It is easy to see that this preserves the desired properties, because p ’s write overwrites and thus “hides” all earlier writes to that register.

Moreover, as p can perform only $o(\log n)$ RMRs (which is larger than the lower bound we are trying to prove), not too many processes need to be removed. As a result, at least a $1/(\log n)^{O(1)}$

fraction of the active processes at the end of round E_{i-1} are still active in round E_i , which is sufficient to obtain the desired lower bound.

The RME lower bound of Chan and Woelfel [4] allows processes to perform, for example, fetch-and-store (FAS) operations, but it does not apply to conventional, non-recoverable mutual exclusion. The reason is exactly the high contention round: If many processes perform a FAS on the same register, then each of them will discover the previous one, and not more than one process can remain active. (In fact, standard constant RMR queue lock algorithms are based on that idea, e.g., [5, 19, 20].) In the recoverable model, however, carefully chosen process crashes can rescue the lower bound idea. Assume that at least $k + 1 = (\log n)^{\Theta(1)}$ processes are poised to perform FAS operations on the same register. Take an arbitrary group of exactly $k + 1$ of those processes, $\alpha, \beta_1, \dots, \beta_k$. For a process β_j from that group consider the following execution: First, let β_j perform its FAS. Then let α perform its FAS (in which it overwrites β_j 's FAS but also learns about β_j), let α crash (so that it forgets about β_j), and finally let α recover and run to completion, which may require removing from the execution $o(\log n)$ other processes that would otherwise be discovered. Unless β_j is one of the processes we have to remove, β_j 's FAS is now hidden by α 's FAS, much in the same way as before earlier writes on a register were hidden by the last write. (The reason why we chose multiple β -processes for the group is that some of them have to be removed because they get discovered by α -processes, possibly from other groups. The size of the group guarantees that sufficiently many groups have at least one β -process that does not get discovered, and can remain active.)

This technique fails again for certain powerful operations, such as fetch-and-add (FAA) with large word-size. This is exactly what Katzan and Morrison [18] exploit: In their algorithm each process p performs $\text{FAA}(2^p)$, effectively setting the p -th bit of the register and learning the set of all processes that performed the operation before p . This makes it impossible to hide any processes.

Intuitively, however, if the memory word size, w , is small enough, then not all processes can leave information in a register. This is what our lower bound exploits. The difficulty is to make no assumption on what kind of information processes can leave behind.

To that end we prove a complex combinatorial lemma (the Process-Hiding Lemma). For simplicity consider a word-size $w = \Theta(\log n)$. Consider a group of $k = (\log n)^{\Theta(1)}$ processes, such that at the beginning of round i all processes in the group are poised to access the same register, and each such access will incur an RMR. As before, we consider a high contention round i when constructing execution E_i from E_{i-1} . A simplified version of the lemma states that we can find two (not necessarily distinct) sets, A and B , of processes in that group, as well as a special process $z \in B$, such that if either exactly each process in A takes a single step (in some predefined order) or exactly each process in B takes a single step, the register value resulting from those steps is the same. In addition, if we let all processes in $A \cup B \setminus \{z\}$ crash, recover, and run to completion, then z will not be discovered. This way, we obtain two possible executions for round i : In the first one, all processes in B (including z) take a single (RMR) step, then all processes in $A \cup B \setminus \{z\}$ crash, recover and run to completion. In the second one, all processes in A take a single step instead of those in B , and the rest is the same as before. Because the register value is the same at the end of each of the two executions, no process except for z can distinguish the two. Thus, if we want to keep z an active process, we use the first execution. If at a later point during our construction (of additional rounds) we realize that we need to remove z from the execution (because otherwise it would get discovered), we can switch to the second one, and no other process is affected by this.

In fact, the statement (and proof) of the Process-Hiding Lemma is more complex than described above, as it has to handle multiple groups simultaneously, each containing k processes that are poised to perform their next step on the same register. Moreover, even though the overall outline of

the entire lower bound proof is similar to [4], the deeper details are quite different and technically much more difficult.

1.2 Other Related Work

Researchers have considered several additional desirable properties of RME algorithms, such as starvation-freedom, first-come-first-serve, or abortability. Below, we focus on the RMR complexity of solutions to the basic RME problem, ignoring any extended properties.

The first RME algorithm by Golab and Ramaraju [11] has an RMR complexity of $O(n)$. This was then improved by Jayanti and Joshi to $O(\log n)$ [15], and finally to $O(\log n / \log \log n)$ by Golab and Hendler [9] for the CC model, and Jayanti, Jayanti, and Joshi [14] for the DSM model. All these algorithms use fetch-and-store, or compare-and-swap, or both. Using fetch-and-increment, Chan and Woelfel obtained constant *amortized* passage complexity [3]. Dhoked and Mittal [6] presented a *crash-adaptive* algorithm, which has improved performance if the total number of process crashes, f , is limited: It has an RMR complexity of $O(\min\{\sqrt{f}, \log n / \log \log n\})$. Katzan and Morrison's algorithm [18] adapts to the number, k , of processes participating concurrently; assuming w -bit fetch-and-add, where $w = \Theta(\log n)$, it has an RMR complexity of $O(\min\{k, \log n / \log \log n\})$.

For the weaker system-wide crash model, where all processes must crash simultaneously, Golab and Hendler [10] solved the RME algorithm with constant RMR complexity. However, the authors assume non-standard system support, guaranteeing that an epoch counter is incremented with each system crash. A recent algorithm by Jayanti, Jayanti, and Joshi [13] also achieves constant RMR complexity without making such assumptions.

2 PRELIMINARIES

We assume the standard asynchronous shared memory model. It comprises n processes with unique IDs and any number of *base objects* (shared memory locations). Processes communicate by performing arbitrary atomic operations (called *steps*) on the base objects. Each base object stores w bits.

We assume (w.l.o.g.) that processes execute *one-time mutual exclusion*: First, a process executes an *entry protocol*. Once completed, the process is in the *critical section*, where it may perform other operations on memory locations that are not used by the mutual exclusion algorithm. Then it executes an *exit protocol*. After that, it takes no more steps. A process that has not yet started the entry protocol or has finished the exit protocol, is in the *remainder section*.

We assume that at any point, when a process is about to perform a step of its entry or exit protocol, it may instead be forced (by the system) to perform a *crash step*. We say the process *crashes* when it performs that step. When that happens, all its local variables are reset to their initial values, and the process immediately begins executing a *recover protocol*. After the recover protocol, it essentially resumes its mutual exclusion protocol; what exactly this means is determined by the RME properties defined below. Generally, RME algorithms also tolerate crash steps in the critical section (see the critical-section reentry property described below), but our lower bound applies regardless.

A process begins a *passage*, when it performs the first shared memory step of its entry or recover protocol, and ends with the next crash step, or when it enters the remainder section (after finishing the exit protocol). A *super-passage* of a process begins with the start of its entry protocol, and ends when the process finishes its subsequent exit protocol. Thus, a super-passage may comprise multiple passages, separated by crash steps, and in the absence of crashes, passages and super-passages are the same.

The algorithm must satisfy *mutual exclusion*, which means that no two processes can be in the critical section at the same time. It must also satisfy *deadlock-freedom*, which requires that some process must eventually enter the remainder section, provided that not all processes are in the remainder section, all processes that are not in the remainder section keep taking steps, and the number of crash steps is finite.

For our lower bound, it is sufficient if each process can crash at most once. The definition of recoverable mutual exclusion [11] also requires another property, called *critical section re-entry*, but our lower bound is independent of that.

To analytically predict the performance of (recoverable and conventional) mutual exclusion algorithms, two models have been studied. In the *cache-coherent* (CC) model, each process is equipped with a cache, and all processes are connected via a bus to the shared memory. Whenever a process performs a read operation it stores a copy of the read value in its cache. Any non-read operation (by any process) of that memory location invalidates the cache copy. We say a process's operation incurs a *remote memory reference* (RMR), if it is a non-read operation, or it is a read operation for a memory location of which the process has no valid cache copy. In the *distributed shared memory* (DSM) model, the shared memory is partitioned into segments, one for each process. An operation by a process p on a shared memory location incurs an RMR if and only if that memory location is not in p 's memory segment. The RMR complexity of a mutual exclusion algorithm is the maximum number of RMRs a process may incur in a passage.

3 THE RME LOWER BOUND PROOF

Consider an arbitrary algorithm that solves the RME problem with $o(w)$ RMR complexity. The goal of this section is to show that this RME algorithm has $\Omega(\min(\log_w n, \log n / \log \log n))$ RMR complexity. Since $\log_w n = \log n / \log \log n$ for $w = \log n$, it suffices to assume that $w \geq \log n$ and then simply show that this RME algorithm has $\Omega(\log_w n)$ RMR complexity. (Clearly, smaller values of w can only increase the RMR complexity.)

Towards that end, we consider a system that uses this RME algorithm in the following manner (all these assumptions can be made w.l.o.g.):

- (A1) The number of processes is sufficiently large such that every passage of every execution incurs no more than w RMRs. (Because of asymptotic notation in the assumptions above, this may not be true for small n .)
- (A2) In the critical section, each process performs exactly one step, which incurs an RMR.
- (A3) Each process crashes at most once.

Then it suffices to construct an execution of this system in which some process never crashes and never enters the critical section, yet incurs $\Omega(\log_w n)$ RMRs.

Let $\mathcal{P} = \{1, \dots, n\}$ be the set of processes in this system, \mathcal{R}' the set of shared objects in the system, and $\mathcal{R} \subseteq \mathcal{R}'$ the set of shared objects used by the RME algorithm. A *configuration* C consists of a state for each process $p \in \mathcal{P}$ and each shared object $R \in \mathcal{R}'$. Let C_0 denote the initial configuration, i.e., the configuration that consists of the initial state of each process and object in this system. A *schedule* is a sequence over $\{p, \hat{p} : p \in \mathcal{P}\}$, where p denotes a non-crash step by process p , and \hat{p} denotes a crash-step by p .

An *execution* is a sequence of *events*, where each event corresponds to a step by some process and contains the following information: the process that is executing the step, the shared memory operation that process is executing, the shared object on which the shared memory operation is executed, and whether the shared memory operation incurs an RMR.

3.1 Proof Strategy

Similar to [1, 4], our proof is constructed in rounds. In each round, every process that is still trying to enter the critical section, takes a number of steps, and incurs at least one RMR. The construction method differs depending on the amount of contention, i.e., whether most processes are poised to access shared objects that many other processes are poised to access. Intuitively, the low contention scenario is simply about removing processes such that no pair of processes can communicate via the same shared object, so its proof is essentially unchanged from earlier results. On the other hand, the high contention scenario requires various techniques to minimize the amount of useful information communicated by processes even when they are accessing the same object. Consequently, our proof significantly differs in the high contention scenario, because unlike these earlier results, our proof needs to deal with arbitrary shared memory operations.

The proof is based on a simple observation: if multiple processes are ‘actively’ attempting to enter the critical section, then they cannot safely enter the critical section before discovering one another, lest they violate mutual exclusion. Thus, throughout the proof several closely related schedules are constructed in a manner that maximizes both the number of these *active* processes and the number of RMRs they incur without discovering one another.

More formally, let $\sigma_{round}[0..\infty][0..2^n - 1]$ be an initially empty table of schedules with an unbounded number of rows and 2^n columns. Roughly speaking, for every non-negative integer i , the i -th row of the table will contain only schedules in which the active processes have incurred at least i RMRs. For every integer $s \in \{0, 1, \dots, 2^n - 1\}$, the s -th column is associated with the unique set $S \subseteq \mathcal{P}$ of processes such that $s = \sum_{p \in S} 2^{p-1}$. Then the s -th column will contain only schedules in which only the processes in S can begin super-passages.

Filling the first row of the table is simple: in the empty schedule, every active process has incurred 0 RMRs, and the set of processes that have begun super-passages is \emptyset , a subset of every possible set of processes. Thus, every cell of $\sigma_{round}[0][0..2^n - 1]$ is set to contain the empty schedule.

The proof then proceeds in rounds, where in each round $i \geq 1$, some cells of the i -th row are filled with schedules derived by appending more steps to the schedules in the $(i - 1)$ -th row. Since the only desired schedules are those in which the active processes do not discover one another, many of the cells in each row will be left with the value \perp , indicating that no schedule matching the required criteria was found. Thus, as the proof iterates through the rows of the table, the number of cells in each row that are filled with schedules decreases.

As such, the goal of each round is to limit this decrease, such that $\Omega(\log_w n)$ rounds complete before the number of schedules becomes too small to continue. After this point, every schedule in the final round would have active processes that incur $\Omega(\log_w n)$ RMRs without entering the critical section (or crashing).

To facilitate that, a number of invariants are maintained on every row of schedules constructed. Roughly speaking, these invariants are:

- (1) The s -th column contains only schedules in which only the processes in the associated set S can begin super-passages.
- (2) There is exactly one *maximal* schedule which has the maximal number of active processes, and all other schedules are ‘sub’-schedules that correspond to every proper subset of the active processes in the maximal schedule. This invariant ensures that if the maximal schedule cannot be extended without allowing some active processes to discover one another, then a sub-schedule can be extended and made into the new maximal schedule for the next round.
- (3) The state of every process is the same in every schedule it is part of. This invariant ensures that the active processes have not discovered one another, since they have the same state in a schedule where there are no other active processes.

- (4) The set of processes that have completed their super-passage is the same over the entire row of schedules. This invariant avoids potentially complicated scenarios where a process that completes its super-passage in one schedule is still active in another.
- (5) For each shared object, its value in each schedule depends only on whether the schedule contains the process that last accessed it in the maximal schedule. This invariant ensures that shared object values are sufficiently similar across different schedules that it becomes difficult for the active processes to later distinguish between different schedules.
- (6) In every schedule, each process crashes at most once, and every process that has not completed its super-passage has never crashed. The invariant limits the number of crashes so that the resulting lower bound holds even when each process crashes at most once.
- (7) In every schedule, every process that has not completed its super-passage has not yet entered the critical section. The invariant makes the proof significantly simpler, since it prevents interactions between the active processes and the inactive processes that have already entered the critical section but not yet completed their super-passage.
- (8) In the DSM model, the shared objects that are owned by active processes have not been accessed by any other active process. This invariant also simplifies the proof, since it prevents non-RMR-incurring steps from allowing an active process to discover another active process, and thus allows the proof to focus on the RMR-incurring steps.
- (9) In the CC model, for each process p , the set of shared objects that p has valid cache copies of is identical over all schedules that contain p . This invariant ensures that in the CC model, for each process p , the number of RMRs incurred by p is the same in every schedule it is part of.
- (10) In the i -th row, every active process in every schedule has incurred at least i RMRs.

It is easy to see that these invariants hold for row 0. Furthermore, for every non-negative integer i , let n_i be the number of active processes in the maximal schedule of row i . Then the second invariant asserts that row i has 2^{n_i} schedules. Moreover, to show that $\Omega(\log_w n)$ rounds can be completed, it suffices to show that for every integer $i \geq 1$, $n_i > n_{i-1}/w^{O(1)}$.

Each round of the proof is divided into two phases: a setup phase in which non-RMR-incurring steps are appended to the schedules until every active process in every schedule is poised to incur an RMR, and a contention phase, in which RMR-incurring steps are appended in specific orders that limit the fraction of active processes discovered.

In the setup phase, multiple non-RMR-incurring step(s) are appended for each active process until they are poised to incur an RMR. By the above invariants, the non-RMR-incurring steps appended for each process are the same in every schedule that contains the process. This is because each process begins with the same state in every schedule that contains the process, and then:

- In the DSM model, its non-RMR-incurring steps only access its own shared objects, which have never been accessed by any other active process, and thus these steps intuitively provide no new information that would cause the process to change its next steps.
- In the CC model, its non-RMR-incurring steps would be reads on shared objects that it has valid cache copies of in every schedule that contains it. Then, since the process already has valid cache copies of these shared objects, they intuitively provide no new information that would cause the process to change its next steps.

In the contention phase, the construction method differs depending on whether at least half of all active processes (in the maximal schedule) are poised to access a shared object that at least k processes are poised to access, where $k = w^d$ for some sufficiently large constant d .

In a low contention scenario, less than half of all active processes (in the maximal schedule) are poised to access a shared object that at least k processes are poised to access. Thus on average, each

shared object has relatively few processes poised to access it. In this case, a graph is constructed with nodes representing the active processes, and edges that intuitively indicate processes that could discover one another: either because they are poised to access the same shared object, or they are poised to access a shared object that is owned or was previously accessed by another active process. Since the contention is relatively low, the resulting graph is relatively sparse, and thus contains a relatively large independent set. Then any schedule that contains any process outside this independent set is discarded, so that the remaining schedules only contain active processes that would not discover one another with their next step. These remaining schedules then have a single step appended for each active process, and then are used to fill the next row of $\sigma_{round}[0..\infty][0..2^n - 1]$. It is straightforward to show that the above invariants still hold for this new row of schedules. Furthermore, due to the relative largeness of the independent set, it is also straightforward to show that $n_i > n_{i-1}/w^{O(1)}$ for every row $i \geq 1$ constructed in a low contention scenario.

In a high contention scenario, at least half of all active processes (in the maximal schedule) are poised to access a shared object that at least k processes are poised to access. Thus, on average, each shared object has relatively many processes poised to access it. In this scenario, it is often inevitable that some active processes are discovered by the others, and these active processes must then be inactivated by allowing them to enter the critical section and then complete their super-passage. To further complicate matters, each such process could discover $O(w)$ other active processes before completing its super-passage, and these discovered processes must then be removed (schedules that contain such processes are discarded).

What makes matters even more complex is that we consider shared objects that support arbitrary operations. Such operations could potentially allow every active process in a single operation to discover all other active processes that have previously accessed the shared object! As such, this is the point where our proof significantly diverges from the earlier results of [1, 4].

First, we divide the active processes into groups of $\Theta(k)$ processes, such that within each group, all processes are poised to access the same shared object (we remove any active processes that cannot be placed into such groups, the number of which is at most a constant fraction of the active processes). We also remove any active processes that either own a shared object that some group is poised to access, or were the last to access a shared object that some group is poised to access. Since this is the high contention scenario, the number of shared objects that the groups are poised to access, and hence the number of active processes that need to be removed, is just a fraction of the total. Then there are two cases: either the majority of groups contain at least one process that is poised to perform a read operation, or not.

The first case is simple: since read operations do not change the state of a shared object, it is impossible to discover processes from the read operations they have performed. Thus we remove any schedule that contains a process that is poised to perform a non-read operation. The remaining schedules then have a single step appended for each remaining active process, and then are used to fill the next row of $\sigma_{round}[0..\infty][0..2^n - 1]$. It is straightforward to show that the above invariants still hold for this new row of schedules. Furthermore, since $k = w^d$ and the majority of groups contain at least one process that is poised to perform a read operation, it is also straightforward to show that $n_i > n_{i-1}/w^{O(1)}$ for every row $i \geq 1$ constructed in a read high contention scenario.

In the second case, we first remove every group that contains a process that is poised to perform a read operation, i.e., we remove any schedule that contains a process from these groups. So all remaining groups contain only active processes that are poised to perform non-read operations. Then let X_1, X_2, \dots, X_m be these remaining groups of active processes, and for each group X_j , let R_j be the shared object that group X_j is poised to access. We then consider certain subsets A_j of

X_j , and associate with each an execution in which each process from A_j takes exactly one step (in a specific fixed order). Thus, for each $A_j \subseteq X_j$ we obtain a unique state of R_j at the end of the associated execution. Let y_j be the state that results from the largest number of subsets of X_j (with ties broken arbitrarily).

At this point, we make another important observation: if there are multiple methods to change a shared object to some state y , any operation that finds the state to be y cannot determine which method was used.

With this in mind, our goal is the following:

- First, identify for every group sets A_j and V_j , where $A_j \subseteq V_j \subseteq X_j$, such that when each process in A_j takes exactly one step (in some predetermined order) the state of R_j is changed to y_j .

The processes in these sets V_j are called the alpha processes, and every schedule that does not contain all of the alpha processes is being discarded. The alpha processes will be crashed to make them forget any information they learned, and then they will be allowed to run until they complete their super-passages. Any other active processes that they discover along the way are removed (schedules that contain such processes are discarded). As a result, these alpha processes are the processes that can potentially be discovered by other processes in future rounds.

- Then, for at least a constant fraction of the groups, identify sets $B_j \subseteq V_j$ as well as a process $z_j \in X_j \setminus V_j$, such that when the processes in B_j take steps in some order, the state of R_j is also changed to y_j . These processes in $B_j \cup \{z_j\}$ are called the beta processes.

Note that A_j and B_j are not necessarily disjoint, or distinct.

This way, we now obtain two possible (sub-)executions: In one, first all processes in A_j take a single step, in the other first all processes in $B_j \cup \{z_j\}$ take a single step. In both executions, after the steps by the processes in A_j or B_j , all processes in V_j crash and then run to completion. The resulting two configurations are indistinguishable to all remaining active processes other than z_j . In other words, the undiscovered active process z_j can take an RMR-incurring step that is “hidden” by the steps of the alpha processes in its group.

Achieving this goal (of finding suitable sets A_j , B_j , and V_j as well as processes z_j), is not trivial. Each alpha process can incur $o(w)$ RMRs in its super-passage, allowing it to discover $o(w)$ active processes. Furthermore, the alpha processes can communicate with one another to change their behaviors, so the set of processes that are discovered by the alpha processes can differ wildly even if only a single alpha process is added or removed, making it difficult to ensure that any chosen process z_j is not discovered.

A new combinatorial lemma, the *Process-Hiding Lemma*, is a key technical contribution of this proof. Essentially, the Process-Hiding Lemma asserts that as long as there are at least $216w^3$ processes within each group, there must exist sets of alpha and beta processes that satisfy the above requirements. (Recall that each group contains $\Theta(k)$ processes, where $k = w^d$, so the lemma is applicable with a sufficiently large constant d .) Then, since $k = w^d$ and a constant fraction of the groups of $\Theta(k)$ processes yield an undiscovered beta process for the new maximal schedule, we can also prove that $n_i > n_{i-1}/w^{O(1)}$ for every row $i \geq 1$ constructed in a high contention scenario.

Thus, regardless of whether each round $i \geq 1$ has a low contention phase or a high contention phase, $n_i > n_{i-1}/w^{O(1)}$. By the second invariant, the number of schedules in each row i is 2^{n_i} . So $\Omega(\log_w n)$ rounds complete before the number of schedules becomes too small to continue. After that the schedules in the final round have active processes that incur $\Omega(\log_w n)$ RMRs without entering the critical section (or crashing). Consequently, the algorithm has $\Omega(\log_w n)$ RMR complexity.

We state and prove the Process-Hiding Lemma in Section 3.2. In Section 3.3, we formally state the invariants that the array of schedules satisfy. Then in Section 3.4, we show how to derive the RMR lower bound from those invariants. The proof of the invariants is deferred to the appendix.

3.2 The Process-Hiding Lemma

In this subsection, we prove the following key statement.

LEMMA 2 (PROCESS-HIDING LEMMA). *Let X and Y be sets, and let $(X_i), i \in \{1, \dots, m\}$, be a partition of X . Let $\ell \geq 0$ be an integer and $\delta \geq 1$ a real number, and suppose that $|Y| \leq 2^\ell$ and $|X_i| \geq 108\delta\ell^2$ for all $i \in \{1, \dots, m\}$. For each $y \in Y$, let $f_y: 2^X \rightarrow Y$ be a function, and let $y_0 \in Y$.*

There exist sequences $(y_i), (A_i)$, and $(V_i), i \in \{1, \dots, m\}$, such that for each $i \in \{1, \dots, m\}$,

$$y_i \in Y \text{ and } \emptyset \subsetneq A_i \subseteq V_i \subseteq X_i,$$

and

- $f_{y_{i-1}}(A_i) = y_i$ for all $i \in \{1, \dots, m\}$;
- for each set $D \subseteq X$ of size $|D| \leq \delta \cdot |\bigcup_{1 \leq i \leq m} V_i|$, there exists a subset of indices $I_D \subseteq \{1, \dots, m\}$ of size $|I_D| \geq m/2$, and sequences (z_i) and $(B_i), i \in I_D$, such that for each $i \in I_D$,

$$z_i \in X_i \setminus (V_i \cup D) \text{ and } B_i \subseteq V_i,$$

$$\text{and } f_{y_{i-1}}(B_i \cup \{z_i\}) = y_i.$$

To understand the power of the Process-Hiding Lemma, first consider the case $m = 1$. Suppose that, at the beginning of a high contention round, X_1 is the set of processes poised to perform a step on the same memory location R , and y_0 is the value of that memory location. Then for a set $A_1 \subseteq X_1$, $f_{y_0}(A_1)$ is the value of R , if all processes in A_1 take one step (in some predefined order). Given sets $A_1 \subseteq V_1 \subseteq X_1$, D is the set of processes that are discovered by the processes in V_1 , if all of them take a crash step and then run to completion. We use δ to indicate the number of shared memory locations a single process can access while running to completion, and thus the number of processes it can discover. The lemma states that there exists a set $B_1 \subseteq V_1$ and a process $z_1 \in X_1 \setminus (V_1 \cup D)$ (which is not being discovered by the processes in V_1), such that if all processes in $B_1 \cup \{z_1\}$ take a step (instead of the processes in A_1), the resulting value of register R is also y_1 . Hence, no matter if the processes in A_1 take a step, or the processes in $B_1 \cup \{z_1\}$, and then all processes in V_1 crash and run to completion, the resulting value of R is the same. Thus, the two configurations are indistinguishable to all remaining processes other than z_1 .

We use $m > 1$ to deal with the case where very many processes are poised to perform an operation on register R . In that case, we split the processes into groups $X_i, i = 1, \dots, m$, each of which is of size roughly w^d for some constant d . For each group X_i we construct sets of processes $V_i \subseteq X_i, A_i \subseteq V_i, B_i \subseteq V_i$, and $z_i \in X_i \setminus V_i$ as before. Now, D is the set of processes discovered, if all processes in $V_1 \cup \dots \cup V_m$ crash and then run to completion. Moreover, y_i is the value the register has after all processes in A_1, A_2, \dots, A_i have taken steps. The lemma states that for a constant fraction of indices $j \in \{1, \dots, m\}$ the same value y_j is obtained, if the processes in A_j are replaced with the processes in $B_j \cup \{z_j\}$.

Note that all variable names used are local to this section. We will use the following operations.

DEFINITION 3. *For any set X , and for any $A \subseteq X$ and $B \subseteq 2^X$, we define*

$$\sigma_A(B) = \{S: S \subseteq B, A \subseteq S\}, \quad \pi_A(B) = \{S \setminus A: S \in \sigma_A(B)\}.$$

If X is a singleton set $X = \{x\}$, we will write σ_x and π_x to denote σ_X and π_X , respectively.

To visualize these definition note that if for any two distinct sets $S, S' \in \sigma_A(B)$ their intersection is $S \cap S' = A$, then $\sigma_A(B)$ is a sunflower system, with core A and petals the elements of $\pi_A(B)$ (see, e.g., [17] for an introduction to sunflower systems). Our analysis, however, does not involve sunflowers or results about them.

Recall that in a k -partite hypergraph $H = (X_1, \dots, X_k, E)$, the vertices are partitioned into k disjoint sets X_1, \dots, X_k , and each hyperedge $e \in E$ contains precisely one vertex from each set. The next basic lemma states that if X_1 has size at most $s \cdot (1 + \epsilon)$, then: either there are two (not necessarily distinct) vertices $z_1, z_2 \in X_1$ such that the set $\pi_{z_1}(E) \cup \pi_{z_2}(E)$ has size at least $|E|/s$; or there are at least $s \cdot (1 + \epsilon)(1 - 2\epsilon)$ distinct vertices $z \in X_1$ such that their sets $\pi_z(E)$ intersect.

LEMMA 4. *Let $H = (X_1, \dots, X_k, E)$ be a k -partite hypergraph such that $|X_1| \leq s \cdot (1 + \epsilon)$, where s is a positive integer and $0 \leq \epsilon < 1/2$. There exists a set $Z \subseteq X_1$ such that*

- (a) $|Z| \leq 2$ and $|\bigcup_{z \in Z} \pi_z(E)| \geq |E|/s$; or
- (b) $|Z| \geq s \cdot (1 + \epsilon)(1 - 2\epsilon)$ and $\bigcap_{z \in Z} \pi_z(E) \neq \emptyset$.

PROOF. We assume that (a) is not true for any $Z \subseteq X_1$, and we will show that (b) holds for some $Z \subseteq X_1$. Let $\ell = |X_1|$, and suppose that $X_1 = \{x_1, \dots, x_\ell\}$. For $1 \leq i \leq \ell$, let $p_i = \pi_{x_i}(E)$, and suppose that $|p_1| \geq |p_2| \geq \dots \geq |p_\ell|$. We can bound $|p_1|$ from below and above as follows. We observe that $\sum_{1 \leq i \leq \ell} |p_i| = |E|$. And since $|p_1| = \max_{1 \leq i \leq \ell} |p_i| \geq (1/\ell) \sum_{1 \leq i \leq \ell} |p_i|$, we have

$$|p_1| \geq |E|/\ell \geq |E|/[s(1 + \epsilon)],$$

since $\ell = |X_1| \leq s(1 + \epsilon)$. Also

$$|p_1| < |E|/s,$$

because of our assumption that (a) is not true for any $Z \subseteq X_1$. Let

$$\lambda = \max \{i : |p_1| + |p_i| \geq |E|/s\}.$$

We now compute a lower bound on $\sum_{1 \leq i \leq \lambda} \frac{|p_1 \cap p_i|}{|p_i|}$. We have

$$\begin{aligned} \sum_{1 \leq i \leq \lambda} \frac{|p_1 \cap p_i|}{|p_i|} &= \sum_{1 \leq i \leq \lambda} \frac{|p_1| + |p_i| - |p_1 \cup p_i|}{|p_i|} \\ &= \sum_{1 \leq i \leq \lambda} \left(1 - \frac{|p_1 \cup p_i| - |p_i|}{|p_i|} \right) \\ &\geq \sum_{1 \leq i \leq \lambda} \left(1 - \frac{|E|/s - |p_i|}{|E|/[s(1 + \epsilon)]} \right) \\ &= \sum_{1 \leq i \leq \lambda} \frac{|E| - |E|(1 + \epsilon) + |p_i|s(1 + \epsilon)}{|E|} \\ &= -\lambda\epsilon + \frac{s(1 + \epsilon)}{|E|} \cdot \sum_{1 \leq i \leq \lambda} |p_i|, \end{aligned}$$

where for the inequality in the third line we used that $|p_1 \cup p_i| < |E|/s$, because of our assumption that (a) does not hold for any $Z \subseteq X_1$, and also that $|p_1| \geq |E|/[s(1 + \epsilon)]$. By the definition of λ , we have $|p_1| + |p_i| < |E|/s$ for $\lambda < i \leq \ell$. Then

$$\sum_{\lambda < i \leq \ell} |p_i| \leq \ell(|E|/s - |p_1|) \leq \ell(|E|/s - |E|/\ell) = (\ell/s - 1)|E| \leq \epsilon|E|,$$

since $\ell = |X_1| \leq s(1 + \epsilon)$. Thus, $\sum_{1 \leq i \leq \lambda} |p_i| \geq (1 - \epsilon)|E|$. Substituting this and $\ell \leq s(1 + \epsilon)$ into the previous equation, gives

$$\sum_{1 \leq i \leq \lambda} \frac{|p_1 \cap p_i|}{|p_1|} \geq -s(1 + \epsilon)\epsilon + s(1 + \epsilon)(1 - \epsilon) = s(1 + \epsilon)(1 - 2\epsilon).$$

We can now use an elementary expectation argument to complete the proof. Choose some $e' \in p_1$ uniformly at random, and let R_i , for $i \in \{1, \dots, \lambda\}$, be the indicator random variable of the event $e' \in p_i$. Then $E[R_i] = \Pr[e' \in p_i] = \frac{|p_1 \cap p_i|}{|p_1|}$, and

$$E \left[\sum_{1 \leq i \leq \lambda} R_i \right] = \sum_{1 \leq i \leq \lambda} \frac{|p_1 \cap p_i|}{|p_1|} \geq s(1 + \epsilon)(1 - 2\epsilon).$$

Since $\sum_{1 \leq i \leq \lambda} R_i$ is the number of sets p_1, \dots, p_λ that contain e' , the above inequality implies that there is some $e^* \in p_1$ that is contained in at least $s(1 + \epsilon)(1 - 2\epsilon)$ sets. This implies that (b) holds for some $Z \subseteq \{x_1, \dots, x_\lambda\} \subseteq X_1$. \square

The next lemma is obtained by iteratively applying [Lemma 4](#). It states that if $|X_i| \leq s \cdot (1 + \epsilon)$ holds for all X_i (not just for X_1), and the total number of hyperedges is $|E| \geq s^k$, then there is a subset of hyperedges and an index d such that: for every $i \neq d$ at most two vertices from X_i are contained in those hyperedges; and at least $s \cdot (1 + \epsilon)(1 - 2\epsilon)$ vertices from X_d are contained in the hyperedges.

LEMMA 5. *Let $H = (X_1, \dots, X_k, E)$ be a k -partite hypergraph such that $|X_i| \leq s \cdot (1 + \epsilon)$ for all $1 \leq i \leq k$, and $|E| \geq s^k$, where s is a positive integer and $0 \leq \epsilon < 1/2$. There exist a set $\{e_1, e_2, \dots\}$ of hyperedges and an index $d \in \{1, \dots, k\}$ such that the set $U = \bigcup_r e_r$ satisfies*

- (a) $|U \cap X_i| \leq 2$ for all $i \neq d$; and
- (b) $|U \cap X_d| \geq s \cdot (1 + \epsilon)(1 - 2\epsilon)$.

PROOF. We recursively construct sequences (Z_i) , $i \in \{1, \dots, d\}$, and (H_i) , $i \in \{0, \dots, d\}$, for some $d \in \{1, \dots, k\}$, such that $Z_i \subseteq X_i$ and $H_i = (X_{i+1}, \dots, X_k, E_i)$ is a $(k - i)$ -partite hypergraph. The index d is also determined by the recursive construction.

We initialize the recursive construction by setting $H_0 = H$. For d we just know that $1 \leq d \leq k$ initially. For each $i \geq 1$, if $i \leq d$ then we determine Z_i and H_i from H_{i-1} as follows. We have two cases.

- Case $i < k$: We let Z_i be a subset of X_i such that:
 - (i) $|Z_i| \leq 2$ and $|\bigcup_{z \in Z_i} \pi_z(E_{i-1})| \geq |E_{i-1}|/s$, or
 - (ii) $|Z_i| \geq s \cdot (1 + \epsilon)(1 - 2\epsilon)$ and $\bigcap_{z \in Z_i} \pi_z(E_{i-1}) \neq \emptyset$.
 From [Lemma 4](#), such a set Z_i exists. Then,
 - If (i) holds, we let $E_i = \bigcup_{z \in Z_i} \pi_z(E_{i-1})$, thus it holds $|E_i| \geq |E_{i-1}|/s$. We also establish that $d > i$.
 - If (ii) holds, we let E_i be a singleton set $E_i = \{e^*\}$ such that $e^* \in \bigcap_{z \in Z_i} \pi_z(E_{i-1})$. We also set $d = i$, and the recursive construction is completed.
- Case $i = k$: We let $Z_i = \bigcup_{e \in E_{i-1}} e \subseteq X_k$, thus $|Z_i| = |E_{i-1}|$. Also we let $E_i = \{\emptyset\}$ and set $d = k$.

It is immediate from the above construction that for $1 \leq i < d$, $|E_i| \geq |E_{i-1}|/s$. And since $|E_0| = |E| \geq s^k$, it follows that for $0 \leq i < d$,

$$|E_i| \geq s^{k-i}.$$

Also, for $1 \leq i < d$,

$$1 \leq |Z_i| \leq 2,$$

where the left inequality holds because $|E_i| \geq s^{k-i} > 0$. For $|Z_d|$, we have that $|Z_d| \geq s \cdot (1+\epsilon)(1-2\epsilon)$ if $d < k$. And if $d = k$ then $|Z_d| = |E_{k-1}| \geq s^{k-(k-1)} = s$. Thus, in all cases

$$|Z_d| \geq s \cdot (1+\epsilon)(1-2\epsilon).$$

Finally, we note that the set E_d is a singleton set, $E_d = \{e^*\}$, where $e^* = \emptyset$ if $d = k$.

Consider now the sequence $E = F_0 \supseteq F_1 \supseteq \dots \supseteq F_d \supseteq F$, where $F_i = \bigcup_{z \in Z_i} \sigma_z(F_{i-1})$ for $1 \leq i \leq d$, and $F = \sigma_{e^*}(F_d)$.¹ It is immediate that F contains all hyperedges $e = \{x_1, x_2, \dots, x_d\} \cup e^* \in E$ such that $x_i \in Z_i$ for $1 \leq i \leq d$. We will show that F is the desired set $\{e_1, e_2, \dots\}$ of hyperedges, i.e., we will show that (a) and (b) hold for $U = \bigcup_{e \in F} e$.

For $1 \leq i \leq d$, let $U_i = \bigcup_{e \in F_i} e$. Since $F_1 \supseteq F_2 \supseteq \dots \supseteq F_d \supseteq F$, it follows $U_1 \supseteq U_2 \supseteq \dots \supseteq U_d \supseteq U$. For $1 \leq i < d$, we have that $U_i \cap X_i \subseteq Z_i$, by the definition of F_i , thus $U \cap X_i \subseteq U_i \cap X_i \subseteq Z_i$, and $|U \cap X_i| \leq |Z_i| \leq 2$. For $d < i \leq k$, we have $U \cap X_i \subseteq e^* \cap X_i$ thus $|U \cap X_i| \leq 1$. We have thus proved (a).

Let $x_d \in Z_d$. Recall that $E_d = \{e^*\}$, where $e^* \in \bigcap_{z \in Z_d} \pi_z(E_{d-1})$ if $d < k$ and $e^* = \emptyset$ otherwise. It follows that $\{x_d\} \cup e^* \in E_{d-1}$. Similarly, by iteratively applying the definition of $E_i = \bigcup_{z \in Z_i} \pi_z(E_{i-1})$, for $i = d-1, d-2, \dots, 1$, we obtain that there is a sequence of vertices $x_{d-1}, x_{d-2}, \dots, x_1$, where $x_i \in Z_i$ for $1 \leq i < d$, such that

$$\{x_1\} \cup \dots \cup \{x_{d-1}\} \cup \{x_d\} \cup e^* \in E_0 = E.$$

Therefore, $\{x_1, \dots, x_d\} \cup e^* \in F$. Since we can repeat the argument for any $x_d \in Z_d$ to obtain a hyperedge in F , it follows $Z_d \subseteq \bigcup_{e \in F} e = U$. Since also $Z_d \subseteq X_d$ and $|Z_d| \geq s \cdot (1+\epsilon)(1-2\epsilon)$, we conclude that

$$|U \cap X_d| \geq |Z_d| \geq s \cdot (1+\epsilon)(1-2\epsilon).$$

This proves (b). □

We now use [Lemma 5](#) to prove our main result.

PROOF OF LEMMA 2. We recursively construct sequences (y_i) , (H_i) , (F_i) , and (d_i) , $i \in \{1, \dots, m\}$, such that $y_i \in Y$, H_i is a hypergraph, F_i is a subset of hyperedges of H_i , and d_i is an integer.

Recall that $y_0 \in Y$ is given in the lemma's statement. For each $i \in \{1, \dots, m\}$, given y_{i-1} , we define y_i , H_i , F_i , and d_i as follows:

Let $k = 4\ell$. Let $X_{i,1}, \dots, X_{i,k}$ be mutually disjoint subsets of X_i , such that for $1 \leq j \leq k$,

$$|X_{i,j}| = \lfloor 27\delta\ell \rfloor.$$

We can define such disjoint sets because $|X_i| \geq 108\delta\ell^2 = k \cdot 27\delta\ell$. Let $(X_{i,1}, \dots, X_{i,k}, E_i)$ be a complete k -partite hypergraph, and for each $y \in Y$, let

$$E_{i,y} = \{e \in E_i : f_{y_{i-1}}(e) = y\}.$$

We let y_i be an element $y \in Y$ that maximizes $|E_{i,y}|$, i.e., $|E_{i,y_i}| \geq |E_{i,y}|$ for all $y \in Y$. We then define H_i to be the k -partite hypergraph $H_i = (X_{i,1}, \dots, X_{i,k}, E_{i,y_i})$. We observe that

$$|E_{i,y_i}| \geq |E_i|/|Y| \geq \lfloor 27\delta\ell \rfloor^k / 2^\ell \geq (\lfloor 27\delta\ell \rfloor / 1.2)^k,$$

¹The definition of sequence (F_i) , $i \in \{0, \dots, d-1\}$, is the same as that of (E_i) , $i \in \{0, \dots, d-1\}$, except that operator σ is used instead of π .

where for the last inequality we used that $\ell = k/4$ and $2^{1/4} < 1.2$. Next, we apply [Lemma 5](#) to hypergraph H_i , for $s = \lfloor 27\delta\ell \rfloor / 1.2$ and $\epsilon = 0.2$. We let F_i be the set of hyperedges and d_i the index predicted by the lemma. Then $\emptyset \neq F_i \subseteq E_{i,y_i}$, $d_i \in \{1, \dots, k\}$, and the set $U_i = \bigcup_{e \in F_i} e$ satisfies²

- (i) $|U_i \cap X_{i,j}| \leq 2$ for all $j \neq d_i$, and
- (ii) $|U_i \cap X_{i,d_i}| \geq s \cdot (1 + \epsilon)(1 - 2\epsilon) = 0.6 \lfloor 27\delta\ell \rfloor$.

We define the sequences (A_i) and (V_i) , $i \in \{1, \dots, m\}$, using the objects constructed above, as follows. For each $i \in \{1, \dots, m\}$, we let A_i be a hyperedge from the set F_i , and let

$$V_i = (U_i \setminus X_{i,d_i}) \cup A_i.$$

Clearly, $\emptyset \subseteq A_i \subseteq V_i \subseteq X_i$. Also, since $A_i \in F_i \subseteq E_{i,y_i}$, we have that $f_{y_{i-1}}(A_i) = y_i$ holds, as desired.

For $1 \leq i \leq m$, we can express V_i as

$$V_i = (U_i \setminus X_{i,d_i}) \cup (A_i \cap X_{i,d_i}),$$

because $A_i \setminus X_{i,d_i} \subseteq U_i \setminus X_{i,d_i}$. Then

$$|V_i| = \sum_{j \neq d_i} |U_i \cap X_{i,j}| + |A_i \cap X_{i,d_i}| \leq 2(k-1) + 1 = 8\ell - 1,$$

where the inequality holds because of (i). Also,

$$|U_i \setminus V_i| = |U_i \cap X_{i,d_i}| - |A_i \cap X_{i,d_i}| \geq 0.6 \lfloor 27\delta\ell \rfloor - 1 \geq 16\delta\ell - 2,$$

where the first inequality holds because of (ii).

Now, let $D \subseteq X$ such that $|D| \leq \delta \cdot |\bigcup_{1 \leq i \leq m} V_i|$. Then

$$|D| \leq \delta \cdot \sum_{1 \leq i \leq m} |V_i| \leq \delta m(8\ell - 1) \leq m(8\delta\ell - 1),$$

since $\delta \geq 1$. Let

$$I_D = \{i \in \{1, \dots, m\} : (U_i \setminus V_i) \setminus D \neq \emptyset\}.$$

To lower-bound the size of I_D we observe that the sets $U_i \setminus V_i$, $1 \leq i \leq m$, are mutually disjoint (because the sets X_i , $1 \leq i \leq m$, are mutually disjoint), and we use inequality $|U_i \setminus V_i| \geq 16\delta\ell - 2$ shown above, to obtain that the number of distinct sets $U_i \setminus V_i$ that are subsets of D is at most

$$\frac{|D|}{16\delta\ell - 2} \leq \frac{m(8\delta\ell - 1)}{16\delta\ell - 2} = \frac{m}{2}.$$

It follows that $|I_D| \geq m - m/2 \geq m/2$, as desired.

Finally we define the sequences (z_i) and (B_i) , $i \in I_D$, as follows. For each $i \in I_D$, let

$$z_i \in (U_i \setminus V_i) \setminus D.$$

Such a z_i exists because $(U_i \setminus V_i) \setminus D \neq \emptyset$, since $i \in I_D$. Also, since $V_i = (U_i \setminus X_{i,d_i}) \cup A_i \supseteq U_i \setminus X_{i,d_i}$, we have

$$(U_i \setminus V_i) \setminus D \subseteq (X_{i,d_i} \setminus V_i) \setminus D = X_{i,d_i} \setminus (V_i \cup D).$$

Thus $z_i \in X_{i,d_i} \setminus (V_i \cup D) \subseteq X_i \setminus (V_i \cup D)$, as desired. Next, for each $i \in I_D$, let $e_i \in F_i$ be hyperedge such that

$$e_i \cap X_{i,d_i} = z_i.$$

Clearly, such a hyperedge e_i exists since $z_i \in U_i \cap X_{i,d_i}$. We let

$$B_i = e_i \setminus \{z_i\}.$$

Then $B_i \subseteq U_i \setminus X_{i,d_i} \subseteq V_i$, as desired. Finally, since $e_i \in F_i \subseteq E_{i,y_i}$, we have $f_{y_{i-1}}(B_i \cup \{z_i\}) = f_{y_{i-1}}(e_i) = y_i$. This concludes the proof of [Lemma 2](#). \square

² $F_i \neq \emptyset$ holds because of (ii).

3.3 Invariants

For every schedule σ , configuration C , and shared object R , we define the following:

- $P(\sigma)$: the set of all processes that have steps in σ .
- $E(C, \sigma)$: the execution determined by σ starting in configuration C .
- $val_R(C, \sigma)$: the value (state) of R at the end of $E(C, \sigma)$.
- $state_p(C, \sigma)$: the state of p at the end of $E(C, \sigma)$.
- $last_R(C, \sigma)$: the process that last performed an operation on R at the end of $E(C, \sigma)$; or \perp if no process has ever performed an operation on R .
- $F(C, \sigma)$: the set of processes that have finished their super-passage at the end of $E(C, \sigma)$.

We also define $E(\sigma) = E(C_0, \sigma)$, $val_R(\sigma) = val_R(C_0, \sigma)$, $state_p(\sigma) = state_p(C_0, \sigma)$, $last_R(\sigma) = last_R(C_0, \sigma)$, and $F(\sigma) = F(C_0, \sigma)$. Then let d be a sufficiently large constant and $k = w^d$. Finally, given any array $A[0..2^n - 1]$ and any set $S \subseteq \mathcal{P}$, we use $A[S]$ to denote $A[\sum_{p \in S} 2^{p-1}]$.

We now formally define the invariants that each array of schedules should satisfy, and detail the construction of these arrays of schedules. Let i be a non-negative integer, and $A[0..2^n - 1]$ be an array such that each array entry contains either a schedule or \perp . Then we say that $A[0..2^n - 1]$ is *i-compliant* if it satisfies the following invariants:

- (I1) For every set $S \subseteq \mathcal{P}$, if $A[S] \neq \perp$, then $P(A[S]) \subseteq S$. (Note that this implies $F(A[S]) \subseteq S$.)
- (I2) There is a unique set $S_{max} \subseteq \mathcal{P}$ such that for every set $S \subseteq \mathcal{P}$, $A[S] \neq \perp$ if and only if $F(A[S_{max}]) \subseteq S \subseteq S_{max}$.
- (I3) For every process $p \in S_{max}$ and every set $S \subseteq \mathcal{P}$ that contains p , if $A[S] \neq \perp$, then $state_p(A[S]) = state_p(A[S_{max}])$.
- (I4) $F(A[S]) = F(A[S_{max}])$ for every set $S \subseteq \mathcal{P}$ with $A[S] \neq \perp$. (Note that this invariant immediately follows from Invariants (I1), (I2), and (I3).)
- (I5) For every shared object $R \in \mathcal{R}$, there is a value y_R such that for every set $S \subseteq \mathcal{P}$, if $A[S] \neq \perp$, then:

$$val_R(A[S]) = \begin{cases} val_R(A[S_{max}]) & \text{if } last_R(A[S_{max}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Note that it is possible that $y_R = val_R(A[S_{max}])$. Furthermore, if $last_R(A[S_{max}]) = \perp \notin S$, then $val_R(A[S]) = y_R$ for every set $S \subseteq \mathcal{P}$ with $A[S] \neq \perp$.

- (I6) For every set $S \subseteq \mathcal{P}$ with $A[S] \neq \perp$, during $E(A[S])$, each process crashes at most once, and each process that is not in $F(A[S])$ never crashes.
- (I7) For every set $S \subseteq \mathcal{P}$ with $A[S] \neq \perp$, each process that is not in $F(A[S])$ does not enter the critical section during $E(A[S])$.
- (I8) In the DSM model, for every process $p \in S_{max} \setminus F(A[S_{max}])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $A[S] \neq \perp$, R can only be accessed by p during $E(A[S])$. (Or equivalently, in the DSM model, for every set $S \subseteq \mathcal{P}$ such that $A[S] \neq \perp$, during $E(A[S])$, each shared object $R \in \mathcal{R}$ can only be accessed by its owner if the owner of R is in $S_{max} \setminus F(A[S_{max}])$.)
- (I9) In the CC model, for every process $p \in S_{max} \setminus F(A[S_{max}])$, there is a set \mathcal{R}_p of shared objects such that for every set $S \subseteq \mathcal{P}$ that contains p , if $A[S] \neq \perp$, then the set of shared objects that p has valid cache copies of at the end of $E(A[S])$ is exactly \mathcal{R}_p . (Or equivalently, for every set $S \subseteq \mathcal{P}$ such that $A[S] \neq \perp$, and every process $p \in S \cap (S_{max} \setminus F(A[S_{max}]))$, the set of shared objects that p has valid cache copies of at the end of $E(A[S])$ is exactly the same as at the end of $E(A[S_{max}])$.)
- (I10) For every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(A[S])$, if $A[S] \neq \perp$, then p incurs at least i RMRs during $E(A[S])$.

Let i be a non-negative integer, and $A[0..2^n - 1]$ be an array that is i -compliant. Then we denote by $S_{\max}(A[0..2^n - 1])$ the unique set of Invariant (I2).

Let $\sigma_{\text{round}}[0..\infty, 0..2^n - 1]$ be a table with all entries initially containing \perp . Our goal is to fill in the table such that for every non-negative integer i , either $\sigma_{\text{round}}[i, 0..2^n - 1]$ is i -compliant, or $i \in \Omega(\log_w n)$.

The proof of the above invariants can be found in [Appendix A](#).

3.4 Completing the Proof of [Theorem 1](#)

For every integer $i \geq 0$, if $\sigma_{\text{round}}[i, 0..2^n - 1]$ is i -compliant, then let $S_{\max}^i = S_{\max}(\sigma_{\text{round}}[i, 0..2^n - 1])$, and let $n_i = |S_{\max}^i \setminus F(\sigma_{\text{round}}[i, S_{\max}^i])|$. We show the following lemma in [Appendix B](#).

LEMMA 6. *For every integer $i \geq 1$, if $\sigma_{\text{round}}[i, 0..2^n - 1]$ is i -compliant, then $n_i \geq n_{i-1}/(64w^{d+1}) - 2$.*

Since $S_{\max}(\sigma_{\text{round}}[0, 0..2^n - 1]) = \mathcal{P}$ and $\sigma_{\text{round}}[0, \mathcal{P}]$ is the empty schedule, $F(\sigma_{\text{round}}[0, \mathcal{P}]) = \emptyset$ and $n_0 = n$. By [Lemma 6](#), for every positive integer i , if $\sigma_{\text{round}}[i, 0..2^n - 1]$ is i -compliant, then $n_i \geq n_{i-1}/O(w^{d+1})$.

Thus, if \mathcal{I} is the largest positive integer such that $\sigma_{\text{round}}[\mathcal{I}, 0..2^n - 1]$ is \mathcal{I} -compliant, then \mathcal{I} is in $\Omega(\log_w n)$. So $\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}]$ contains a schedule such that:

- Since we reach the \mathcal{I} -th iteration, $\sigma_{\text{round}}[\mathcal{I} - 1, 0..2^n - 1]$ has at least $2^{(k^3)}$ non- \perp entries, i.e., $n_{\mathcal{I}-1} \geq k^3 = w^{3d}$. So $n_{\mathcal{I}} \geq w^{3d}/(64w^{d+1}) - 2$, which since d is sufficiently large constant, $n_{\mathcal{I}} \geq w^d$. Thus $|S_{\max}^{\mathcal{I}} \setminus F(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])| \geq w^d$.
- For every process $p \in S_{\max}^{\mathcal{I}} \setminus F(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$, p incurs at least \mathcal{I} RMRs during $E(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$ (Invariant (I10)).
- For every process $p \in S_{\max}^{\mathcal{I}} \setminus F(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$, p never crashes during $E(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$ (Invariant (I6)).
- For every process $p \in S_{\max}^{\mathcal{I}} \setminus F(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$, p never enters the critical section during $E(\sigma_{\text{round}}[\mathcal{I}, S_{\max}^{\mathcal{I}}])$ (Invariant (I7)).

Thus we have proven [Theorem 1](#).

4 CONCLUSION

In this paper we proved tight lower bounds for recoverable mutual exclusion, showing a tradeoff between the word-size, w , of atomic shared base objects and the worst-case RMR complexity. These are the first such lower bounds that are independent of the type of atomic shared memory operations the system provides. The lower bounds are tight for $w = (\log n)^{\Omega(1)}$, matching an upper bound of Katzan and Morrison [18]. For $w = (\log n)^{o(1)}$ it is not known if our RMR complexity lower bound of $\Omega(\log n / \log \log n)$ is tight—we are not aware of any RME algorithm that works with such small shared memory words.

Our lower bound is for the maximum number of RMRs a process incurs during a passage. It is most likely not possible to extend this result to amortized RMR complexity (which measures the average number of RMRs per passage in a worst-case execution). This is due to an RME algorithm with $O(1)$ amortized RMR complexity by Chan and Woelfel [3]. While this algorithm uses unbounded fetch-and-increment objects, it seems quite possible that these can be replaced with bounded ones.

Jayanti, Jayanti, and Joshi [13] presented an RME algorithm with constant RMR complexity in the *system-wide* failure model, where all processes can only crash at the same time. (See also [10] for an algorithm that requires some specialized OS support.) While this algorithm uses an unbounded fetch-and-increment object, it seems that it may be possible to bound it. This would mean that our lower bound, which inherently relies on individual process crashes, cannot be generalized to the system-wide failure model.

It may also be interesting to explore if randomization can help circumvent our impossibility result.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments. Support is gratefully acknowledged from the Agence Nationale de la Recherche (ANR) under project ByBloS (ANR-20-CE25-0002), the Natural Science and Engineering Research Council of Canada (NSERC) under Discovery Grant RGPIN/2019-04852, and the Canada Research Chairs program.

REFERENCES

- [1] James H. Anderson and Yong-Jik Kim. 2002. An Improved Lower Bound for the Time Complexity of Mutual Exclusion. *Distributed Computing* 15 (2002), 221–253.
- [2] Hagit Attiya, Danny Hendler, and Philipp Woelfel. 2008. Tight RMR Lower Bounds for Mutual Exclusion and Other Problems. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*. 217–226.
- [3] David Yu Cheng Chan and Philipp Woelfel. 2020. Recoverable Mutual Exclusion with Constant Amortized RMR Complexity from Standard Primitives. In *Proceedings of the 2020 ACM Symposium on Principles of Distributed Computing (PODC)*. 181–190. <https://doi.org/10.1145/3382734.3405736>
- [4] David Yu Cheng Chan and Philipp Woelfel. 2021. Tight Lower Bound for the RMR Complexity of Recoverable Mutual Exclusion. In *Proceedings of the 40th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*. 533–543. <https://doi.org/10.1145/3465084.3467938>
- [5] Travis Craig. 1993. Building FIFO and Priority-Queueing Spin Locks from Atomic Swap. Technical Report TR-93-02-02, Department of Computer Science, University of Washington.
- [6] Sahil Dhoked and Neeraj Mittal. 2020. An Adaptive Approach to Recoverable Mutual Exclusion. In *Proceedings of the 2020 ACM Symposium on Principles of Distributed Computing (PODC)*. 1–10. <https://doi.org/10.1145/3382734.3405739>
- [7] E. W. Dijkstra. 1965. Solution of a Problem in Concurrent Programming Control. *Commun. ACM* 8 (1965), 569.
- [8] Wojciech Golab, Vassos Hadzilacos, Danny Hendler, and Philipp Woelfel. 2012. RMR-Efficient Implementations of Comparison Primitives Using Read and Write Operations. *Distributed Computing* 25, 2 (2012), 109–162. <https://doi.org/10.1007/s00446-011-0150-8>
- [9] Wojciech Golab and Danny Hendler. 2017. Recoverable Mutual Exclusion in Sub-logarithmic Time. In *Proceedings of the 2017 ACM Symposium on Principles of Distributed Computing (PODC)*. 211–220. <https://doi.org/10.1145/3087801.3087819>
- [10] Wojciech Golab and Danny Hendler. 2018. Recoverable Mutual Exclusion Under System-Wide Failures. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing (PODC)*. 17–26. <https://doi.org/10.1145/3212734.3212755>
- [11] Wojciech Golab and Aditya Ramaraju. 2019. Recoverable mutual exclusion. *Distributed Computing* 32, 6 (2019), 535–564. <https://doi.org/10.1007/s00446-019-00364-0>
- [12] Prasad Jayanti, Siddhartha Jayanti, and Anup Joshi. 2018. Optimal Recoverable Mutual Exclusion Using only FASAS. In *Networked Systems (NETYS) - 6th International Conference*. 191–206. https://doi.org/10.1007/978-3-030-05529-5_13
- [13] Prasad Jayanti, Siddhartha Jayanti, and Anup Joshi. 2023. Constant RMR Recoverable Mutex under System-wide Crashes. arXiv:2302.00748 <https://doi.org/10.48550/arXiv.2302.00748>
- [14] Prasad Jayanti, Siddhartha V. Jayanti, and Anup Joshi. 2019. A Recoverable Mutex Algorithm with Sub-logarithmic RMR on Both CC and DSM. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing (PODC)*. 177–186. <https://doi.org/10.1145/3293611.3331634>
- [15] Prasad Jayanti and Anup Joshi. 2017. Recoverable FCFS Mutual Exclusion with Wait-Free Recovery. In *Proceedings of the 31st International Symposium on Distributed Computing (DISC)*. 30:1–30:15. <https://doi.org/10.4230/LIPIcs.DISC.2017.30.2225-2252>.
- [16] Prasad Jayanti and Anup Joshi. 2022. Recoverable Mutual Exclusion with Abortability. *Computing* 104, 10 (2022), 2225–2252. <https://doi.org/10.1007/s00607-022-01105-1>
- [17] Stasys Jukna. 2011. *Extremal combinatorics: With applications in computer science*. Springer. <https://doi.org/10.1007/978-3-642-17364-6>
- [18] Daniel Katzman and Adam Morrison. 2020. Recoverable, Abortable, and Adaptive Mutual Exclusion with Sublogarithmic RMR Complexity. In *Proceedings of 24th International Conference On Principles Of Distributed Systems (OPODIS)*. 15:1–15:16. <https://doi.org/10.4230/LIPIcs.OPODIS.2020.15>
- [19] Peter Magnusson, Anders Landin, and Erik Hagersten. 1994. Queue Locks on Cache Coherent Multiprocessors. In *Proc. of 8th International Symposium on Parallel Processing*. 165–171. <https://doi.org/10.1109/IPPS.1994.288305>
- [20] John M. Mellor-Crummey and Michael L. Scott. 1991. Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors. *ACM Transactions on Computer Systems* 9, 1 (1991), 21–65.

- [21] Aditya Ramaraju. 2015. *RGLock: Recoverable mutual exclusion for non-volatile main memory systems*. Master's thesis. University of Waterloo. <https://uwaterloo.ca/handle/10012/9473>
- [22] Jae-Heon Yang and James H. Anderson. 1995. A Fast, Scalable Mutual Exclusion Algorithm. *Distributed Computing* 9, 1 (1995), 51–60.

APPENDIX

A PROOF OF THE INVARIANTS

A.1 Base Case

For every set $S \subseteq \mathcal{P}$, let $\sigma_{\text{round}}[0, S]$ be set to the empty schedule (so every entry of $\sigma_{\text{round}}[0, 0..2^n - 1]$ is the empty schedule). Clearly, the array $\sigma_{\text{round}}[0, 0..2^n - 1]$ is 0-compliant with $S_{\text{max}}(\sigma_{\text{round}}[0, 0..2^n - 1]) = \mathcal{P}$ and has 2^n non- \perp entries.

We now iterate through $i = 1, 2, \dots$ as follows:

A.2 Termination Phase

If $\sigma_{\text{round}}[i - 1, 0..2^n - 1]$ is not $(i - 1)$ -compliant or has less than $2^{(k^3)}$ non- \perp entries, terminate.

A.3 Setup Phase

For every set $S \subseteq \mathcal{P}$, let $\sigma_{\text{old}}[S] = \sigma_{\text{round}}[i - 1][S]$. Note that the array $\sigma_{\text{old}}[0..2^n - 1]$ is $(i - 1)$ -compliant; otherwise we would have terminated in this iteration.

Thus by Invariant (I2), there is a unique set $S_{\text{max}}^{\text{old}} \subseteq \mathcal{P}$ such that $\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}] \neq \perp$ and for every set $S \subseteq \mathcal{P}$, $\sigma_{\text{old}}[S] \neq \perp$ if and only if $F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}]) \subseteq S \subseteq S_{\text{max}}^{\text{old}}$. So by definition, $S_{\text{max}}(\sigma_{\text{old}}[0..2^n - 1]) = S_{\text{max}}^{\text{old}}$. Then for every process $p \in S_{\text{max}}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$, let $S_p = \{p\} \cup F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$. Note that $F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}]) \subseteq S_p \subseteq S_{\text{max}}^{\text{old}}$, so $\sigma_{\text{old}}[S_p] \neq \perp$.

Now for every process $p \in S_{\text{max}}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$, let σ_p be a schedule consisting only of the maximum non-negative number of non-crash steps of p such that any RMRs incurred by p in $E(\sigma_{\text{old}}[S_p] \circ \sigma_p)$ were also incurred in $E(\sigma_{\text{old}}[S_p])$. Then let C_p be the configuration at the end of $E(\sigma_{\text{old}}[S_p])$. So by definition, p does not incur any RMRs in $E(C_p, \sigma_p)$. Furthermore, if σ_p is finite, then an RMR would be incurred by p at the end of $E(\sigma_{\text{old}}[S_p] \circ \sigma_p \circ p)$.

LEMMA 7. *For every process $p \in S_{\text{max}}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$, σ_p is finite.*

PROOF. Suppose, for contradiction, that σ_p is infinite for some process $p \in S_{\text{max}}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$.

Recall that $S_p = \{p\} \cup F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$, so $F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}]) \subseteq S_p \subseteq S_{\text{max}}^{\text{old}}$. Since $\sigma_{\text{old}}[0..2^n - 1]$ is $i - 1$ -compliant with $S_{\text{max}}(\sigma_{\text{old}}[0..2^n - 1]) = S_{\text{max}}^{\text{old}}$, by Invariants (I2) and (I4), $\sigma_{\text{old}}[S_p] \neq \perp$ and $F(\sigma_{\text{old}}[S_p]) = F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$. So $p \in S_p \setminus F(\sigma_{\text{old}}[S_p])$. Thus by Invariant (I7), p does not enter the critical section during $E(\sigma_{\text{old}}[S_p])$.

By definition, C_p is the configuration at the end of $E(\sigma_{\text{old}}[S_p])$ and p does not incur any RMRs in $E(C_p, \sigma_p)$. So by (A2), p does not enter the critical section during the infinite execution $E(\sigma_{\text{old}}[S_p] \circ \sigma_p)$.

Now recall that $S_p = \{p\} \cup F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}])$, so every process $q \neq p$ is either in $F(\sigma_{\text{old}}[S_p])$ or not in S_p . By Invariant (I1), every process $q \notin S_p$ takes no steps in $E(\sigma_{\text{old}}[S_p])$. Since σ_p contains only steps of p and $p \in S_p$, every process $q \notin S_p$ also takes no steps in $E(\sigma_{\text{old}}[S_p] \circ \sigma_p)$. So in $E(\sigma_{\text{old}}[S_p] \circ \sigma_p)$, p takes infinitely many steps without entering the critical section while every process $q \neq p$ is in the remainder section — contradicting that $E(\sigma_{\text{old}}[S_p] \circ \sigma_p)$ is an execution of an algorithm that solves the RME problem. \square

Since $\sigma_{\text{old}}[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\text{max}}(\sigma_{\text{old}}[0..2^n - 1]) = S_{\text{max}}^{\text{old}}$, by Invariant (I2), for every set $S \subseteq \mathcal{P}$, $\sigma_{\text{old}}[S] \neq \perp$ if and only if $F(\sigma_{\text{old}}[S_{\text{max}}^{\text{old}}]) \subseteq S \subseteq S_{\text{max}}^{\text{old}}$. So for each set $S \subseteq \mathcal{P}$ such

that $\sigma_{old}[S] \neq \perp$, let $p_{1,S}$ be the process with the smallest ID in $S \setminus F(\sigma_{old}[S_{max}^{old}])$, $p_{2,S}$ be the process with the second smallest ID, and so on. Then let C_S be the configuration at the end of $E(\sigma_{old}[S])$, and let $\sigma_S = \sigma_{p_{1,S}} \circ \sigma_{p_{2,S}} \circ \dots$. Note that by Lemma 7 and the fact that the system has only a finite number of processes, σ_S is a finite schedule.

LEMMA 8. For every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$:

(S1) No RMRs are incurred during $E(C_S, \sigma_S)$.

(S2) For each process $p \in S$, $state_p(C_p, \sigma_p) = state_p(C_S, \sigma_S)$.

(S3) For each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p incurs an RMR at the end of $E(C_S, \sigma_S \circ p)$.

(S4) For each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p has not left the critical section at the end of $E(C_S, \sigma_S)$.

(S5) $F(\sigma_{old}[S]) = F(\sigma_{old}[S] \circ \sigma_S)$.

(S6) In the DSM model, each shared object $R \in \mathcal{R}$ can only be accessed by its owner during $E(C_S, \sigma_S)$.

(S7) In the DSM model, for each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ and each shared object $R \in \mathcal{R}$ owned by p , $val_R(C_p, \sigma_p) = val_R(C_S, \sigma_S)$.

(S8) In the CC model, each shared object $R \in \mathcal{R}$ can only be read during $E(C_S, \sigma_S)$.

(S9) In the CC model, during $E(C_S, \sigma_S)$, each process p can only read shared objects that it already has valid cache copies of.

PROOF. Let $S \subseteq \mathcal{P}$ be any set of processes such that $\sigma_{old}[S] \neq \perp$. Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_{old}[0..2^n - 1]) = S_{max}^{old}$, by Invariant (I2), for every set $S' \subseteq \mathcal{P}$, $\sigma_{old}[S'] \neq \perp$ if and only if $F(\sigma_{old}[S_{max}^{old}]) \subseteq S' \subseteq S_{max}^{old}$. Thus $F(\sigma_{old}[S_{max}^{old}]) \subseteq S \subseteq S_{max}^{old}$. Then since $S \subseteq S_{max}^{old}$, by definition we have that for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, $S_p = \{p\} \cup F(\sigma_{old}[S_{max}^{old}])$.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, by Invariant (I3), for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ and every process $q \in S_p$, $state_q(\sigma_{old}[S]) = state_q(\sigma_{old}[S_p])$. Furthermore, by Invariant (I4), $F(\sigma_{old}[S_{max}^{old}]) = F(\sigma_{old}[S])$. So by Invariant (I7), every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not entered the critical section during $E(\sigma_{old}[S])$.

The proof now differs depending on the model:

CC Model: In the CC model, any step that does not incur an RMR must be a read operation on a shared object that the invoking process already has a valid cache copy of. Thus, by definition, for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p only performs read operations on shared objects that it already has valid cache copies of during $E(C_p, \sigma_p)$.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, by Invariant (I9), for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, the set of shared objects that p has valid cache copies of is in same in C_p as in C_S . Furthermore, read operations clearly cannot invalidate any valid cache copies. Thus, by the definition of σ_S , observe that for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, the operations performed by p during $E(C_S, \sigma_S)$ are the same as in during $E(C_p, \sigma_p)$, i.e., p only performs read operations on shared objects that it already has valid cache copies of during $E(C_S, \sigma_S)$ (S9).

This implies the following:

- Since RMRs are not incurred by any read operation on a shared object that the invoking process already has a valid cache copy of, no RMRs are incurred during $E(C_S, \sigma_S)$ (S1).
- Since only read operations are performed during $E(C_S, \sigma_S)$, each shared object can only be read during $E(C_S, \sigma_S)$ (S8).
- By definition, σ_S contains only steps of processes in $S \setminus F(\sigma_{old}[S_{max}^{old}])$. Thus for every process $p \in S$, observe that $state_p(C_S, \sigma_S) = state_p(C_p, \sigma_p)$ (S2).

Now recall that for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, the set of shared objects that p has valid cache copies of is in same in C_p as in C_S . Then, since (i) $state_p(C_S, \sigma_S) = state_p(C_p, \sigma_p)$, (ii) the valid cache copies of p are the same in C_p as in C_S , (iii) new cache copies cannot be created by reading shared objects that valid cache copies already exist for, and (iv) p incurs an RMR in $E(C_p \circ \sigma_p, p)$ by the definition of σ_p , observe that p also incurs an RMR at the end of $E(C_S, \sigma_S \circ p)$ (S3).

- Since every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not entered the critical section during $E(\sigma_{old}[S])$ and no RMRs are incurred during $E(C_S, \sigma_S)$, by (A2), every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not left the critical section at the end of $E(C_S, \sigma_S)$ (S4).

Then, since σ_S contains only non-crash steps, no process completes during $E(C_S, \sigma_S)$.

Thus $F(\sigma_{old}[S]) = F(\sigma_{old}[S] \circ \sigma_S)$ (S5).

DSM Model: In the DSM model, any step that does not incur an RMR must be an operation on a shared object owned by the invoking process. Thus, by definition, for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p only performs operations on its own shared objects during $E(C_p, \sigma_p)$. Since $\sigma_{old}[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_{old}[0..2^n - 1]) = S_{max}^{old}$ by Invariant (I8), for every process $p \in S_{max}^{old} \setminus F(\sigma_{old}[S_{max}^{old}])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S' \subseteq \mathcal{P}$ with $\sigma_{old}[S'] \neq \perp$, R can only be accessed by p during $E(\sigma_{old}[S'])$, so $last_R(\sigma_{old}[S'])$ is either p or \perp . Thus by Invariant (I5), if $init_R$ is the initial value of R , then:

$$val_R(\sigma_{old}[S']) = \begin{cases} val_R(\sigma_{old}[S_{max}^{old}]) & \text{if } p \in S' \\ init_R & \text{otherwise} \end{cases}$$

So for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ and every shared object $R \in \mathcal{R}$ owned by p , $val_R(\sigma_{old}[S]) = val_R(\sigma_{old}[S_p])$. Furthermore, operations on shared objects not owned by p clearly cannot change the value of shared objects owned by p . Consequently, by the definition of σ_S , observe that for every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, the operations performed by p during $E(C_S, \sigma_S)$ are the same as in during $E(C_p, \sigma_p)$, i.e., p only performs operations on its own shared objects during $E(C_S, \sigma_S)$.

This implies the following:

- Since RMRs are not incurred by any operation on a shared object owned by the invoking process, no RMRs are incurred during $E(C_S, \sigma_S)$ (S1).
- Since each process only accesses its own shared objects during $E(C_S, \sigma_S)$, for each shared object $R \in \mathcal{R}$, R can only be accessed by its owner during $E(C_S, \sigma_S)$ (S6).

Furthermore, since the operations performed by each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ during $E(C_S, \sigma_S)$ are the same as in during $E(C_p, \sigma_p)$, $val_R(C_p, \sigma_p) = val_R(C_S, \sigma_S)$ (S7).

- By definition, σ_S contains only steps of processes in $S \setminus F(\sigma_{old}[S_{max}^{old}])$. Thus for every process $p \in S$, observe that $state_p(C_S, \sigma_S) = state_p(C_p, \sigma_p)$ (S2).

Furthermore, by the definition of σ_p , p incurs an RMR at the end of $E(C_p, \sigma_p \circ p)$, i.e., p is poised to access a shared object that it does not own at the end of $E(C_p, \sigma_p)$. Thus, since $state_p(C_S, \sigma_S) = state_p(C_p, \sigma_p)$, p is also poised to access a shared object that it does not own at the end of $E(C_S, \sigma_S)$, and so p also incurs an RMR at the end of $E(C_S, \sigma_S \circ p)$ (S3).

- Since every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not entered the critical section during $E(\sigma_{old}[S])$ and no RMRs are incurred during $E(C_S, \sigma_S)$, by (A2), every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not left the critical section at the end of $E(C_S, \sigma_S)$ (S4).

Then, since every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not left the critical section at the end of $E(C_S, \sigma_S)$, every process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$ has not completed its super-passage at the end of $E(C_S, \sigma_S)$. Thus $F(\sigma_{old}[S]) = F(\sigma_{old}[S] \circ \sigma_S)$ (S5).

□

We now construct a new array $\sigma_A^S[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] = \perp$ if $\sigma_{old}[S] = \perp$; otherwise $\sigma_A^S[S] = \sigma_{old}[S] \circ \sigma_S$.

LEMMA 9. *Except for Invariant (I7), $\sigma_A^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_A^S[0..2^n - 1]) = S_{max}^{old}$.*

PROOF. For every set $S \subseteq \mathcal{P}$, if $\sigma_A^S[S] \neq \perp$, then by construction, $\sigma_A^S[S] = \sigma_{old}[S] \circ \sigma_S$. Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, by Invariant (I1), $P(\sigma_{old}[S]) \subseteq S$. By the definition of σ_S , σ_S contains only steps of processes in S . Thus $P(\sigma_A^S[S]) \subseteq S$ (Invariant (I1)).

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_{old}[0..2^n - 1]) = S_{max}^{old}$, by Invariant (I2), for every set $S \subseteq \mathcal{P}$, $\sigma_{old}[S] \neq \perp$ if and only if $F(\sigma_{old}[S_{max}^{old}]) \subseteq S \subseteq S_{max}^{old}$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] = \perp$ if and only if $\sigma_{old}[S] = \perp$. Furthermore, by Lemma 8 (S5), $F(\sigma_{old}[S_{max}^{old}]) = F(\sigma_A^S[S_{max}^{old}])$. Thus for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] \neq \perp$ if and only if $F(\sigma_A^S[S_{max}^{old}]) \subseteq S \subseteq S_{max}^{old}$ (Invariant (I2)).

By Lemma 8 (S2), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, and every process $p \in S$, $state_p(\sigma_A^S[S_p]) = state_p(C_p, \sigma_p) state_p(C_S, \sigma_S) = state_p(\sigma_A^S[S])$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] = \perp$ if and only if $\sigma_{old}[S] = \perp$. Furthermore, we have already proven that for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] \neq \perp$ if and only if $F(\sigma_A^S[S_{max}^{old}]) \subseteq S \subseteq S_{max}^{old}$. Thus observe that for every process $p \in S_{max}^{old}$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^S[S] \neq \perp$, then $state_p(\sigma_A^S[S]) = state_p(\sigma_A^S[S_{max}^{old}])$ (Invariant (I3)).

Since we have already proven that Invariants (I1), (I2), and (I3) hold for $\sigma_A^S[0..2^n - 1]$, it immediately follows that Invariant (I4) also holds.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, Invariant (I6) holds for $\sigma_{old}[0..2^n - 1]$. For every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, σ_S contains no crash steps. Thus Invariant (I6) also holds for $\sigma_A^S[0..2^n - 1]$.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, Invariant (I8) holds for $\sigma_{old}[0..2^n - 1]$. By Lemma 8 (S6), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, each shared object $R \in \mathcal{R}$ can only be accessed by its owner during $E(C_S, \sigma_S)$. Thus Invariant (I8) also holds for $\sigma_A^S[0..2^n - 1]$.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, Invariant (I9) holds for $\sigma_{old}[0..2^n - 1]$. In the CC model, by Lemma 8 (S8), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, each shared object $R \in \mathcal{R}$ can only be read during $E(C_S, \sigma_S)$. Thus no valid cache copy can be invalidated during $E(C_S, \sigma_S)$. Furthermore, by Lemma 8 (S8), during $E(C_S, \sigma_S)$, each process p can only read shared objects that it already has valid cache copies of. Thus no new cache copies can be created during $E(C_S, \sigma_S)$. Consequently, Invariant (I9) also holds for $\sigma_A^S[0..2^n - 1]$.

Since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, Invariant (I10) holds for $\sigma_{old}[0..2^n - 1]$. Then, since no steps are removed in the construction of the schedules for $\sigma_A^S[0..2^n - 1]$, clearly Invariant (I10) also holds for $\sigma_A^S[0..2^n - 1]$.

We will now prove that Invariant (I5) holds for $\sigma_A^S[0..2^n - 1]$ as follows. Let $R \in \mathcal{R}$ be any shared object. Our goal is to show that there exists a value y_R such that for every set $S \subseteq \mathcal{P}$, if $\sigma_A^S[S] \neq \perp$, then:

$$val_R(\sigma_A^S[S]) = \begin{cases} val_R(\sigma_A^S[S_{max}^{old}]) & \text{if } last_R(\sigma_A^S[S_{max}^{old}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Note that since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_{old}[0..2^n - 1]) = S_{max}^{old}$, by Invariant (I5), there is a value y_R such that for every set $S \subseteq \mathcal{P}$, if $\sigma_{old}[S] \neq \perp$, then:

$$val_R(\sigma_{old}[S]) = \begin{cases} val_R(\sigma_{old}[S_{max}^{old}]) & \text{if } last_R(\sigma_{old}[S_{max}^{old}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

First, suppose that R is not accessed during $E(C_S, \sigma_S)$ for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$. Then since $\sigma_{old}[S] = \perp$ if and only if $\sigma_A^S[S] = \perp$, and R is not accessed during $E(C_{S_{max}^{old}}, \sigma_{S_{max}^{old}})$, $last_R(\sigma_{old}[S_{max}^{old}]) = last_R(\sigma_A^S[S_{max}^{old}])$. Thus as we wanted, for every set $S \subseteq \mathcal{P}$, if $\sigma_A^S[S] \neq \perp$, then:

$$val_R(\sigma_A^S[S]) = \begin{cases} val_R(\sigma_A^S[S_{max}^{old}]) & \text{if } last_R(\sigma_A^S[S_{max}^{old}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

So suppose instead that there exists a set $S' \subseteq \mathcal{P}$ such that $\sigma_{old}[S'] \neq \perp$, and a process $p \in S' \setminus F(\sigma_{old}[S_{max}^{old}])$ that accesses R during $E(C_{S'}, \sigma_{S'})$. The proof now differs depending on the model.

In the DSM model, by [Lemma 8 \(S6\)](#), p must be the owner of R . Then since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, by [Invariant \(I8\)](#), either $last_R(\sigma_{old}[S_{max}^{old}]) = p$ or $val_R(\sigma_{old}[S_{max}^{old}]) = y_R$. Furthermore, by [Lemma 8 \(S6\)](#) and [\(S7\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, if $p \in S$, then $val_R(C_S, \sigma_S) = val_R(C_p, \sigma_p)$; otherwise $val_R(C_S, \sigma_S) = y_R$. Thus as we wanted, for every set $S \subseteq \mathcal{P}$, if $\sigma_A^S[S] \neq \perp$, then:

$$val_R(\sigma_A^S[S]) = \begin{cases} val_R(\sigma_A^S[S_{max}^{old}]) & \text{if } last_R(\sigma_A^S[S_{max}^{old}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Finally, in the CC model, by [Lemma 8 \(S9\)](#), p already has a valid cache copy of R in $C_{S'}$. So since $\sigma_{old}[0..2^n - 1]$ is $(i-1)$ -compliant, by [Invariant \(I9\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$ and $p \in S$, $val_R(\sigma_{old}[S]) = val_R(\sigma_{old}[S'])$. Thus either $last_R(\sigma_{old}[S_{max}^{old}]) = p$ or $val_R(\sigma_{old}[S_{max}^{old}]) = y_R$. Note that if any process other than p also accesses R during $E(C_S, \sigma_S)$, then $val_R(\sigma_{old}[S_{max}^{old}]) = y_R$. Furthermore, by [Lemma 8 \(S8\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_{old}[S] \neq \perp$, $val_R(\sigma_{old}[S]) = val_R(\sigma_A^S[S])$. Thus observe that as we wanted, for every set $S \subseteq \mathcal{P}$, if $\sigma_A^S[S] \neq \perp$, then:

$$val_R(\sigma_A^S[S]) = \begin{cases} val_R(\sigma_A^S[S_{max}^{old}]) & \text{if } last_R(\sigma_A^S[S_{max}^{old}]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Consequently we have proven that [Invariant \(I5\)](#) holds for $\sigma_A^S[0..2^n - 1]$. \square

We now construct another array $\sigma_B^S[0..2^n - 1]$ with the goal of satisfying [Invariant \(I7\)](#) as follows. If no process is within the critical section at the end of $E(\sigma_A^S[S_{max}^{old}])$, we simply construct $\sigma_B^S[0..2^n - 1]$ such that $\sigma_B^S[0..2^n - 1] = \sigma_A^S[0..2^n - 1]$. Furthermore, we define $S_{max}^S = S_{max}^{old}$.

Otherwise, to avoid violating mutual exclusion, there must be exactly one process $p \in S_{max}^{old} \setminus F(\sigma_A^S[S_{max}^{old}])$ such that at the end of $E(\sigma_A^S[S_{max}^{old}])$, p is within the critical section. Then note that by [Lemma 9](#) and [Invariant \(I3\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^S[S] \neq \perp$, if $p \in S$ then p is also within the critical section at the end of $E(\sigma_A^S[S])$; otherwise no process is within the critical section at the end of $E(\sigma_A^S[S])$. Thus we construct $\sigma_B^S[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_B^S[S] = \perp$; otherwise $\sigma_B^S[S] = \sigma_A^S[S]$. Furthermore, we define $S_{max}^S = S_{max}^{old} \setminus \{p\}$.

LEMMA 10. *This new array $\sigma_B^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(S_{max}^S[0..2^n - 1]) = S_{max}^S$.*

PROOF. By [Lemma 9](#), except for [Invariant \(I7\)](#), $\sigma_A^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^{old}$.

If no process is within the critical section at the end of $E(\sigma_A^S[S_{max}^{old}])$, then $\sigma_B^S[0..2^n - 1] = \sigma_A^S[0..2^n - 1]$. Thus by [Lemma 9](#) and [Lemma 8 \(S4\)](#), [Invariant \(I7\)](#) also holds for $\sigma_A^S[0..2^n - 1]$, and so it follows that $\sigma_B^S[0..2^n - 1] = \sigma_A^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S = S_{max}^{old}$.

Otherwise, there is exactly one process $p \in S_{max}^{old} \setminus F(\sigma_A^S[S_{max}^{old}])$ such that p is within the critical section at the end of $E(\sigma_A^S[S_{max}^{old}])$, and for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_B^S[S] = \perp$; otherwise $\sigma_B^S[S] = \sigma_A^S[S]$. By [Lemma 9](#), except for [Invariant \(I7\)](#), $\sigma_A^S[0..2^n - 1]$ is $(i-1)$ -compliant with

$S_{max}(\sigma_A^S[0..2^n - 1]) = S_{max}^{old}$. Thus observe that by the construction of $\sigma_B^S[0..2^n - 1]$, Invariants (I1), (I2), (I3), (I4), (I5), (I6), (I8), (I9), (I10) must all also hold for $\sigma_B^S[0..2^n - 1]$ with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S = S_{max}^{old} \setminus \{p\}$.

By the construction of $\sigma_B^S[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_B^S[S] = \perp$. Thus for every set $S \subseteq \mathcal{P}$, if $\sigma_B^S[S] \neq \perp$, then $p \notin S$. Consequently, for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^S[S] \neq \perp$, no process is within the critical section at the end of $E(\sigma_A^S[S]) = E(\sigma_B^S[S])$. Therefore by Lemma 8 (S4), Invariant (I7) holds for $\sigma_B^S[0..2^n - 1]$, and thus $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. \square

A.4 Decision Phase

Then for each shared object $R \in \mathcal{R}$, let B_R be the set of processes poised to access R at the end of $E(\sigma_B^S[S_{max}^S])$. Thus $B_R \cap F(\sigma_B^S[S_{max}^S]) = \emptyset$ for each shared object R . Furthermore:

$$S_{max}^S \setminus F(\sigma_B^S[S_{max}^S]) = \bigcup_{R \in \mathcal{R}} B_R$$

We now define:

$$H = \bigcup_{R \in \mathcal{R}, |B_R| \geq k} B_R$$

$$L = \bigcup_{R \in \mathcal{R}, |B_R| < k} B_R$$

Thus H , L , and $F(\sigma_B^S[S_{max}^S])$ are mutually disjoint sets such that:

$$S_{max}^S = H \cup L \cup F(\sigma_B^S[S_{max}^S])$$

Finally, if $|L| \geq |H|$, we end this i -th iteration with a low contention phase; otherwise we end this i -th iteration with a high contention phase.

A.5 Low Contention Case: $|L| \geq |H|$

We begin by constructing an undirected graph where the processes in L are the nodes, and for every pair of nodes p and q , we connect an edge between p and q if and only if at least one of the following is true at the end of $E(\sigma_B^S[S_{max}^S])$:

- p and q are poised to access the same shared object.
- p is poised to access a shared object owned by q , or vice versa.
- p is poised to access a shared object that q has previously performed an operation on, or vice versa.

Thus in this graph:

- Since $|B_R| < k$ for every shared object $R \in \mathcal{R}$ that the processes in L are poised to access, there are at most $k|L|$ edges representing pairs of processes that are poised to access the same shared object: at most k edges for each process in L , connecting it to the other processes that are poised to access the same shared object.
- Since every shared object $R \in \mathcal{R}$ is owned by at most one process, there are at most $|L|$ edges representing processes that are poised to access a shared object owned by some process in L : at most one edge for each process in L , connecting it to the owner of the shared object it is poised to access.
- By (A1), every passage incurs at most w RMRs. By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant. Recall that $L \cap F(\sigma_B^S[S_{max}^S]) = \emptyset$. So by Invariant (I6), each process in L has never crashed, and so has started at most one passage. Thus each process in L has performed

operations on at most w shared objects that are not owned by itself. (Note that shared objects owned by itself are excluded because their associated edges would have already been added.)

Then since $|B_R| < k$ for every shared object $R \in \mathcal{R}$ that processes in L are poised to access, there are at most $wk|L|$ edges representing processes that are poised to access a shared object that some process in L has previously performed an operation on: (k edges from each of the w shared objects previously accessed by each process in L).

Hence, the total number of edges is at most $k|L| + |L| + wk|L| < 3wk|L|$.

Let I be the maximum independent set of the graph.

LEMMA 11. $|I| \geq |L|/(7wk)$.

PROOF. Since there are at most $3wk|L|$ edges, the average degree of the graph is at most $6wk$. The lemma immediately follows by Turan's Theorem. \square

Let $S_I = F(\sigma_B^S[S_{max}^S]) \cup I$. Note that since $I \subseteq L$, $F(\sigma_B^S[S_{max}^S]) \subseteq S_I \subseteq S_{max}^S$. We now construct a new array $\sigma_A^L[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_I$, then $\sigma_A^L[S] = \perp$; otherwise $\sigma_A^L[S] = \sigma_B^S[S]$.

LEMMA 12. *This new array $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^L[0..2^n - 1]) = S_I$.*

PROOF. By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. By construction, $\sigma_A^L[0..2^n - 1]$ is simply a modification of $\sigma_B^S[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains any process in $S_{max}^S \setminus S_I$ has had $\sigma_A^L[S]$ set to \perp , where $F(\sigma_B^S[S_{max}^S]) \subseteq S_I \subseteq S_{max}^S$. It suffices to observe that every invariant still holds with $S_{max}(\sigma_A^L[0..2^n - 1]) = S_I$, and thus $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^L[0..2^n - 1]) = S_I$. \square

Now for each set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, let C'_S be the configuration at the end of $E(\sigma_A^L[S])$, and let σ'_S be the schedule consisting of exactly one non-crash step by each process in $S \setminus F(\sigma_A^L[S_I])$ in order from the process with the smallest ID to the process with the largest ID. Note that σ'_S is a finite schedule since there are only a finite number of processes in the system. Also note that by Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^L[0..2^n - 1]) = S_I$, so $\sigma_A^L[S] \neq \perp$ if and only if $F(\sigma_A^L[S_I]) \subseteq S \subseteq S_I$. Thus σ'_S contains exactly one non-crash step of each process in $I \cap S$ and no other steps.

LEMMA 13. *For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$:*

- (L1) *For each shared object $R \in \mathcal{R}$, R is accessed by at most one process $p \in I \cap S$ during $E(C'_S, \sigma'_S)$ and no other processes.*
- (L2) *For each process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then the owner of R is not in $I \setminus \{p\}$.*
- (L3) *For each process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then R has never been accessed by any process in $I \setminus \{p\}$ during $E(\sigma_A^L[S])$.*
- (L4) *For each process $p \in I$, during $E(C'_S, \sigma'_S)$, p cannot invalidate any cache copy of any process in $I \setminus \{p\}$.*
- (L5) *For every process $p \in I$, if p accesses a shared object R during $E(C'_{S_I}, \sigma'_{S_I})$, then there is a value y_R such that for every set $S' \subseteq \mathcal{P}$, if $\sigma_A^L[S'] \neq \perp$, then:*

$$\text{val}_R(\sigma_A^L[S']) = \begin{cases} \text{val}_R(\sigma_A^L[S_I]) & \text{if } p \in S' \\ y_R & \text{otherwise} \end{cases}$$

Note that this implies that for each shared object $R \in \mathcal{R}$, if R is accessed during $E(C'_S, \sigma'_S)$ then $\text{val}_R(\sigma_A^L[S_I]) = \text{val}_R(\sigma_A^L[S])$.

(L6) For each shared object $R \in \mathcal{R}$, if R is accessed during $E(C'_S, \sigma'_S)$ then $\text{val}_R(C'_{S_I}, \sigma'_{S_I}) = \text{val}_R(C'_S, \sigma'_S)$.

(L7) For each process $p \in S$, $\text{state}_p(C'_{S_I}, \sigma'_{S_I}) = \text{state}_p(C'_S, \sigma'_S)$.

(L8) Each process in $I \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$.

(L9) For each process $p \in S \setminus F(\sigma_A^L[S_I])$, p has not left the critical section during $E(\sigma_A^L[S] \circ \sigma'_S)$.

(L10) $F(\sigma_A^L[S]) = F(\sigma_A^L[S] \circ \sigma'_S)$.

PROOF. Let $S \subseteq \mathcal{P}$ be any set of processes such that $\sigma_A^L[S] \neq \perp$. By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. So by Invariant (I2), for every set $S' \subseteq \mathcal{P}$, $\sigma_A^L[S'] \neq \perp$ if and only if $F(\sigma_A^L[S_I]) \subseteq S' \subseteq S_I$. Thus $F(\sigma_A^L[S_I]) \subseteq S \subseteq S_I$.

Furthermore, by Invariants (I3) and (I4), for every process $p \in S$, $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_A^L[S_I])$ and $F(\sigma_A^L[S]) = F(\sigma_A^L[S_I])$. So by Invariant (I7), every process $p \in S \setminus F(\sigma_A^L[S_I])$ has not entered the critical section during $E(\sigma_A^L[S])$.

Now recall that σ'_S contains exactly one non-crash step of each process in $I \cap S$ and no other steps. Also recall that by construction, for every process $p \in S$, $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_A^L[S_I]) = \text{state}_p(\sigma_B^S[S_I]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$. Then since I is an independent set of the graph we constructed earlier:

- No pair of processes in I are poised to access the same shared object at the end of $E(\sigma_B^S[S_{\max}^S])$. Since $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$ for every process $p \in S$, no pair of processes in I are poised to access the same shared object at the end of $E(\sigma_A^L[S])$. Thus every process in $I \cap S$ accesses a different shared object during $E(C'_S, \sigma'_S)$ (L1).
- No process $p \in I$ is poised to access a shared object owned by a different process $q \in I$ at the end of $E(\sigma_B^S[S_{\max}^S])$. Since $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$ for every process $p \in S$, no process $p \in I$ is poised to access a shared object owned by a different process $q \in I$ at the end of $E(\sigma_A^L[S])$. Thus for each process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then the owner of R is not in $I \setminus \{p\}$. (L2).
- No process $p \in I$ is poised to access a shared object that has previously been accessed by a different process $q \in I$ at the end of $E(\sigma_B^S[S_{\max}^S])$. Since $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$ for every process $p \in S$, no process $p \in I$ is poised to access a shared object that has previously been accessed by a different process $q \in I$ at the end of $E(\sigma_A^L[S])$. Thus for each process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then R has never been accessed by any process in $I \setminus \{p\}$ during $E(\sigma_A^L[S])$. (L3). Therefore, for each process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, no process in $I \setminus \{p\}$ makes a cache copy of R during $E(\sigma_A^L[S])$. Consequently, for each process $p \in I$, during $E(C'_S, \sigma'_S)$, p cannot invalidate any cache copy of any process in $I \setminus \{p\}$ (L4).

Furthermore, since $\text{state}_p(\sigma_A^L[S_I]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$ for every process $p \in S$, no process $p \in I$ is poised to access a shared object that has previously been accessed by a different process $q \in I$ at the end of $E(\sigma_A^L[S_I])$. Thus for every process $p \in I$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then $\text{last}_R(\sigma_A^L[S_I]) \notin I \setminus \{p\}$, so either $\text{last}_R(\sigma_A^L[S_I]) = p$ or $\text{last}_R(\sigma_A^L[S_I]) \in F(\sigma_A^L[S_I]) \cup (\mathcal{P} \setminus S_I)$.

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. Thus by Invariant (I5), for every shared object $R \in \mathcal{R}$, there is a value y_R such that for every set

$S' \subseteq \mathcal{P}$, if $\sigma_A^L[S'] \neq \perp$, then:

$$\text{val}_R(\sigma_A^L[S']) = \begin{cases} \text{val}_R(\sigma_A^L[S_I]) & \text{if } \text{last}_R(\sigma_A^L[S_I]) \in S' \\ y_R & \text{otherwise} \end{cases}$$

We just showed that for every process $p \in I$, if p accesses a shared object R during $E(C'_{S_I}, \sigma'_{S_I})$, then either $\text{last}_R(\sigma_A^L[S_I]) = p$ or $\text{last}_R(\sigma_A^L[S_I]) \in F(\sigma_A^L[S_I]) \cup (\mathcal{P} \setminus S_I)$. Thus for every process $p \in I$, if p accesses a shared object R during $E(C'_{S_I}, \sigma'_{S_I})$, then there is a value y_R such that for every set $S' \subseteq \mathcal{P}$, if $\sigma_A^L[S'] \neq \perp$, then:

$$\text{val}_R(\sigma_A^L[S']) = \begin{cases} \text{val}_R(\sigma_A^L[S_I]) & \text{if } p \in S' \\ y_R & \text{otherwise} \end{cases}$$

Consequently, observe that for each shared object $R \in \mathcal{R}$, if R is accessed during $E(C'_S, \sigma'_S)$ then $\text{val}_R(\sigma_A^L[S_I]) = \text{val}_R(\sigma_A^L[S])$ (L5).

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. So by Invariant (I3), for each process $p \in S$, $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_A^L[S_I])$. Then, since we have proven (L1) and (L5), during both $E(C'_S, \sigma'_S)$ and $E(C'_{S_I}, \sigma'_{S_I})$, for each process $p \in I \cap S$, p performs the same operation on the same shared object, causing the same resulting state and response. This implies that:

- For each shared object $R \in \mathcal{R}$, if R is accessed during $E(C'_S, \sigma'_S)$ then $\text{val}_R(C'_{S_I}, \sigma'_{S_I}) = \text{val}_R(C'_S, \sigma'_S)$ (L6).
- For each process $p \in S$, $\text{state}_p(C'_{S_I}, \sigma'_{S_I}) = \text{state}_p(C'_S, \sigma'_S)$ (L7). (Recall that σ'_S consists of exactly one non-crash step of each process in $I \cap S$ and no other steps, so the states of other processes do not change.)

Since $\sigma_A^L[S] \neq \perp$, by construction, $\sigma_{\text{old}}[S] \neq \perp$. By Lemma 8 (S3), for each process $p \in S \setminus F(\sigma_{\text{old}}[S_{\max}^{\text{old}}])$, p incurs an RMR at the end of $E(\sigma_A^S[S] \circ p)$. By construction and Lemma 9 (Invariant (I4)), $\sigma_A^L[S] = \sigma_B^S[S] = \sigma_A^S[S]$, and $F(\sigma_A^L[S_I]) = F(\sigma_A^S[S_I]) = F(\sigma_A^S[S_{\max}^{\text{old}}]) = F(\sigma_{\text{old}}[S_{\max}^{\text{old}}])$. Thus for each process $p \in S \setminus F(\sigma_A^L[S_I])$, p incurs an RMR at the end of $E(\sigma_A^L[S] \circ p)$. By definition, $S_I = I \cup F(\sigma_A^L[S_I])$, so since $S \subseteq S_I$, $S \setminus F(\sigma_A^L[S_I]) = I \cap S$. Finally, recall that σ'_S contains exactly one non-crash step of each process in $I \cap S$ and no other steps. Consequently, since we have already proven that each process $I \cap S$ accesses a different shared object during $E(C'_S, \sigma'_S)$, observe that each process in $I \cap S$ must incur exactly one RMR during $E(C'_S, \sigma'_S)$ (L8).

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i-1)$ -compliant, so by Invariant (I7), each process that is not in $F(\sigma_A^L[S])$ does not enter the critical section during $E(\sigma_A^L[S])$. Recall that σ'_S consists of exactly one non-crash step of each process in $I \cap S = S \setminus F(\sigma_A^L[S_I])$ and no other steps. Thus, with only one step, although a process could enter the critical section during $E(C'_S, \sigma'_S)$, it cannot have taken any steps within the critical section. Thus by (A2), no process can leave the critical section during $E(C'_S, \sigma'_S)$. So for each process $p \in S \setminus F(\sigma_A^L[S_I])$, p has not left the critical section during $E(\sigma_A^L[S] \circ \sigma'_S)$ (L9). Therefore, since no process in $S \setminus F(\sigma_A^L[S_I])$ has left the critical section during $E(\sigma_A^L[S] \circ \sigma'_S)$, no process in $S \setminus F(\sigma_A^L[S_I])$ has completed its super-passage during $E(\sigma_A^L[S] \circ \sigma'_S)$. Thus $F(\sigma_A^L[S]) = F(\sigma_A^L[S] \circ \sigma'_S)$ (L10). \square

We now construct a new array $\sigma_B^L[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] = \perp$ if $\sigma_A^L[S] = \perp$; otherwise $\sigma_B^L[S] = \sigma_A^L[S] \circ \sigma'_S$.

LEMMA 14. *Except for Invariant (I7), $\sigma_B^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_B^L[0..2^n - 1]) = S_I$.*

PROOF. For every set $S \subseteq \mathcal{P}$, if $\sigma_B^L[S] \neq \perp$, then by construction, $\sigma_B^L[S] = \sigma_A^L[S] \circ \sigma'_S$. By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I1), $P(\sigma_A^L[S]) \subseteq S$. By the definition of σ'_S , σ'_S contains only steps of processes in $S \cap I$. Thus $P(\sigma_B^L[S]) \subseteq S$ (Invariant (I1)).

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. So by Invariant (I2), for every set $S \subseteq \mathcal{P}$, $\sigma_A^L[S] \neq \perp$ if and only if $F(\sigma_A^L[S_I]) \subseteq S \subseteq S_I$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] = \perp$ if and only if $\sigma_A^L[S] = \perp$. Furthermore, by Lemma 13 (L10), $F(\sigma_B^L[S_I]) = F(\sigma_A^L[S_I])$. Thus for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] \neq \perp$ if and only if $F(\sigma_B^L[S_I]) \subseteq S \subseteq S_I$ (Invariant (I2)).

By Lemma 13 (L7), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, and every process $p \in S$, $\text{state}_p(\sigma_B^L[S_I]) = \text{state}_p(C'_{S_I}, \sigma'_{S_I}) \text{state}_p(C'_S, \sigma'_S) = \text{state}_p(\sigma_B^L[S])$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] = \perp$ if and only if $\sigma_A^L[S] = \perp$. Furthermore, we have already proven that for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] \neq \perp$ if and only if $F(\sigma_B^L[S_I]) \subseteq S \subseteq S_I$. Thus for every process $p \in S_I$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_B^L[S] \neq \perp$, then $\text{state}_p(\sigma_B^L[S]) = \text{state}_p(\sigma_B^L[S_I])$ (Invariant (I3)).

Since we have already proven that Invariants (I1), (I2), and (I3) hold for $\sigma_B^L[0..2^n - 1]$, it immediately follows that Invariant (I4) also holds.

By Lemma 13 (L5), for every process $p \in I$, if p accesses a shared object R during $E(C'_{S_I}, \sigma'_{S_I})$, then there is a value y_R such that for every set $S' \subseteq \mathcal{P}$, if $\sigma_A^L[S'] \neq \perp$, then:

$$\text{val}_R(\sigma_A^L[S']) = \begin{cases} \text{val}_R(\sigma_A^L[S_I]) & \text{if } p \in S' \\ y_R & \text{otherwise} \end{cases}$$

Note that since p accesses R during $E(C'_{S_I}, \sigma'_{S_I})$, by Lemma 13 (L1), $\text{last}_R(\sigma_B^L[S_I]) = p$. By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant, so if $p \in S'$, then p also accesses R during $E(C'_{S'}, \sigma'_{S'})$. Thus by Lemma 13 (L6), if $p \in S'$, then $\text{val}_R(\sigma_B^L[S']) = \text{val}_R(\sigma_B^L[S_I])$. Furthermore, by Lemma 13 (L1), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, R cannot be accessed by any process other than p during $E(C'_{S'}, \sigma'_{S'})$. Therefore if $p \notin S'$, then R cannot be accessed during $E(C'_{S'}, \sigma'_{S'})$, and so $\text{val}_R(\sigma_B^L[S']) = y_R$. Thus we have that for every shared object $R \in \mathcal{R}$ such that some process p accesses R during $E(C'_{S_I}, \sigma'_{S_I})$, for every set $S' \subseteq \mathcal{P}$, if $\sigma_B^L[S'] \neq \perp$, then:

$$\text{val}_R(\sigma_B^L[S']) = \begin{cases} \text{val}_R(\sigma_B^L[S_I]) & \text{if } p = \text{last}_R(\sigma_B^L[S_I]) \in S' \\ y_R & \text{otherwise} \end{cases}$$

Then, since any shared object that is not accessed by any process during $E(C'_{S_I}, \sigma'_{S_I})$ clearly does not change its state, observe that Invariant (I5) holds for $\sigma_B^L[0..2^n - 1]$.

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant, so Invariant (I6) holds for $\sigma_A^L[0..2^n - 1]$. For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, σ'_S contains no crash steps. Thus Invariant (I6) also holds for $\sigma_B^L[0..2^n - 1]$.

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. So by Invariant (I8), for every process $p \in S_I \setminus F(\sigma_A^L[S_I])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $\sigma_A^L[S] \neq \perp$, R can only be accessed by p during $E(\sigma_A^L[S])$. By Lemma 13 (L2), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, for each process $p \in I$, if p accesses a shared object R during $E(C'_{S'}, \sigma'_{S'})$, then the owner of R is not in $I \setminus \{p\} = (S_I \setminus F(\sigma_A^L[S_I])) \setminus \{p\}$. In other words, for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, for each process $p \in I$, if p owns a shared object R , then no other process in $I = S_I \setminus F(\sigma_A^L[S_I])$ accesses R during $E(C'_{S'}, \sigma'_{S'})$. By Lemma 13 (L10), $F(\sigma_A^L[S_I]) = F(\sigma_B^L[S_I])$. Consequently, for every process $p \in S_I \setminus F(\sigma_B^L[S_I])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $\sigma_B^L[S] \neq \perp$, R can only be accessed by p during $E(\sigma_B^L[S])$ (Invariant (I8)).

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_A^L[0..2^n - 1]) = S_I$. So by Invariant (I3), for every process $p \in S_I$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^L[S] \neq \perp$, then $\text{state}_p(\sigma_A^L[S]) = \text{state}_p(\sigma_A^L[S_I])$. Furthermore, by Invariant (I9), in the CC model, for every process $p \in S_I \setminus F(\sigma_A^L[S_I])$, there is a set \mathcal{R}_p of shared objects such that for every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^L[S] \neq \perp$, then the set of shared objects that p has valid cache copies of at the end of $E(\sigma_A^L[S])$ is exactly \mathcal{R}_p . Recall that for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, σ_S' contains exactly one non-crash step of each process in $I \cap S$ and no other steps. By Lemma 13 (L1), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, each shared object is accessed by at most one process $p \in I \cap S$ during $E(C_S', \sigma_S')$ and no other processes. Furthermore, by Lemma 13 (L5), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, for every shared object $R \in \mathcal{R}$, if R is accessed during $E(C_S', \sigma_S')$, then $\text{val}_R(\sigma_A^L[S_I]) = \text{val}_R(\sigma_A^L[S])$. Thus for every process $p \in S_I$ and every set $S \subseteq \mathcal{P}$ such that S contains p and $\sigma_A^L[S] \neq \perp$, during both $E(C_S', \sigma_S')$ and $E(C_{S_I}', \sigma_{S_I}')$, p performs the same operation on the same shared object which begins with the same value, causing the same resulting state and response. There are two cases: either this operation that p performs on R is a read operation, or it is not.

If it is a read operation, then no cache copies are invalidated by the read, and p creates a new valid cache copy of R during both $E(C_S', \sigma_S')$ and $E(C_{S_I}', \sigma_{S_I}')$, thus observe that for every set $S' \subseteq \mathcal{P}$ that contains p , if $\sigma_B^L[S'] \neq \perp$, then the set of shared objects that p has valid cache copies of is exactly $\mathcal{R}_p \cup \{R\}$. If it is not a read operation, then cache copies can be invalidated, but by Lemma 13 (L4), the invalidated cache copies cannot belong to any process in $I \setminus \{p\} = (S_I \setminus F(\sigma_A^L[S_I])) \setminus \{p\}$. Thus the cache copies of every process in $(S_I \setminus F(\sigma_A^L[S_I])) \setminus \{p\}$ are unaffected, whereas observe that for every set $S' \subseteq \mathcal{P}$ that contains p , if $\sigma_B^L[S'] \neq \perp$, then the set of shared objects that p has valid cache copies of is exactly $\mathcal{R}_p \setminus \{R\}$. By Lemma 13 (L10), $F(\sigma_A^L[S_I]) = F(\sigma_B^L[S_I])$. Consequently, in both cases, for every process $p \in S_I \setminus F(\sigma_B^L[S_I])$, there is a set \mathcal{R}'_p of shared objects (namely either $\mathcal{R}_p \cup \{R\}$ or $\mathcal{R}_p \setminus \{R\}$ where R is the one shared object that p is poised to access at the end of $E(\sigma_A^L[S_I])$) such that for every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_B^L[S] \neq \perp$, then the set of shared objects that p has valid cache copies of is exactly \mathcal{R}'_p (Invariant (I9)).

By Lemma 12, $\sigma_A^L[0..2^n - 1]$ is $(i-1)$ -compliant, so by Invariant (I10), for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_A^L[S])$, if $\sigma_A^L[S] \neq \perp$, then p incurs at least $i-1$ RMRs during $E(\sigma_A^L[S])$. By Lemma 13 (L8), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^L[S] \neq \perp$, each process in $I \cap S$ incurs exactly one RMR during $E(C_S', \sigma_S')$. By construction, for every set $S \subseteq \mathcal{P}$, if $\sigma_B^L[S] \neq \perp$, then $\sigma_B^L[S] = \sigma_A^L[S] \circ \sigma_S'$, i.e., every process in $I \cap S = S \setminus F(\sigma_A^L[S_I])$ incurs exactly one more RMR during $E(\sigma_B^L[S])$ than during $E(\sigma_A^L[S])$. By Lemma 13 (L10), $F(\sigma_A^L[S_I]) = F(\sigma_B^L[S_I])$. Since we have already proven that Invariant (I4) holds for $\sigma_B^L[0..2^n - 1]$, $F(\sigma_B^L[S_I]) = F(\sigma_B^L[S])$ for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^L[S] \neq \perp$. Thus for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_B^L[S])$, if $\sigma_B^L[S] \neq \perp$, then p incurs at least i RMRs during $E(\sigma_A^L[S])$ (Invariant (I10)). \square

We now construct another array $\sigma_C^L[0..2^n - 1]$ with the goal of satisfying Invariant (I7) as follows. If no process is within the critical section at the end of $E(\sigma_B^L[S_I])$, we simply construct $\sigma_C^L[0..2^n - 1]$ such that $\sigma_C^L[0..2^n - 1] = \sigma_B^L[0..2^n - 1]$. Furthermore, we define $S_{\max}^L = S_I$.

Otherwise, to avoid violating mutual exclusion, there must be exactly one process $p \in S_I \setminus F(\sigma_B^L[S_I])$ such that at the end of $E(\sigma_B^L[S_I])$, p is within the critical section. Then note that by Lemma 14 and Invariant (I3), for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^L[S] \neq \perp$, if $p \in S$ then p is also within the critical section at the end of $E(\sigma_B^L[S])$; otherwise no process is within the critical section at the end of $E(\sigma_B^L[S])$. Thus we construct $\sigma_C^L[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^L[S] = \perp$; otherwise $\sigma_C^L[S] = \sigma_B^L[S]$. Furthermore, we define $S_{\max}^L = S_I \setminus \{p\}$.

LEMMA 15. *This new array $\sigma_C^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_C^L[0..2^n - 1]) = S_{\max}^L$.*

PROOF. By Lemma 14, except for Invariant (I7), $\sigma_B^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_B^L[0..2^n - 1]) = S_I$.

If no process is within the critical section at the end of $E(\sigma_B^L[S_I])$, then $\sigma_C^L[0..2^n - 1] = \sigma_B^L[0..2^n - 1]$. Thus by Lemma 13 (L9) and Lemma 14, Invariant (I7) also holds for $\sigma_C^L[0..2^n - 1]$, and so it follows that $\sigma_C^L[0..2^n - 1] = \sigma_B^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_C^L[0..2^n - 1]) = S_{\max}^L = S_I$.

Otherwise, there is exactly one process $p \in S_I \setminus F(\sigma_B^L[S_I])$ such that p is within the critical section at the end of $E(\sigma_B^L[S_I])$, and for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^L[S] = \perp$; otherwise $\sigma_C^L[S] = \sigma_B^L[S]$. By Lemma 14, except for Invariant (I7), $\sigma_B^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_B^L[0..2^n - 1]) = S_I$. Thus observe that by the construction of $\sigma_C^L[0..2^n - 1]$, Invariants (I1), (I2), (I3), (I4), (I5), (I6), (I8), (I9), (I10) must all also hold for $\sigma_C^L[0..2^n - 1]$ with $S_{\max}(\sigma_C^L[0..2^n - 1]) = S_{\max}^L = S_I \setminus \{p\}$.

By the construction of $\sigma_C^L[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^L[S] = \perp$. Thus for every set $S \subseteq \mathcal{P}$, if $\sigma_C^L[S] \neq \perp$, then $p \notin S$. Consequently, for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^L[S] \neq \perp$, no process is within the critical section at the end of $E(\sigma_C^L[S]) = E(\sigma_B^L[S])$. Therefore by Lemma 13 (L9), Invariant (I7) holds for $\sigma_C^L[0..2^n - 1]$, and thus $\sigma_C^L[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_C^L[0..2^n - 1]) = S_{\max}^L$. \square

Finally, we terminate this i -th iteration by setting $\sigma_{\text{round}}[i, 0..2^n - 1] = \sigma_C^L[0..2^n - 1]$.

A.6 High Contention Case: $|L| < |H|$

By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_B^S[0..2^n - 1]) = S_{\max}^S$. Recall that for every shared object $R \in \mathcal{R}$, B_R is the set of processes poised to access R at the end of $E(\sigma_B^S[S_{\max}^S])$, and:

$$H = \bigcup_{R \in \mathcal{R}, |B_R| \geq k} B_R$$

We first divide the processes in H into groups of exactly k processes such that within each group, all processes are poised to access the same shared object at the end of $E(\sigma_B^S[S_{\max}^S])$. We make as many such groups as possible. Then let H_1 be the set of processes in the resulting groups, i.e., H_1 is a modification of H where all processes that are not in any group are removed. By this construction and the definition of H , we derive that $|H_1| > |H|/2$.

Next, let H_2 be a modification of H_1 such that for each process $p \in H_1$, p is in H_2 if and only if both of the following are true:

- No process in H_1 is poised to access a shared object owned by p at the end of $E(\sigma_B^S[S_{\max}^S])$.
- No process in H_1 is poised to access a shared object R at the end of $E(\sigma_B^S[S_{\max}^S])$ such that $\text{last}_R(\sigma_B^S[S_{\max}^S]) = p$.

Note that since H_1 is composed of groups of exactly k processes that are all poised to access the same shared object at the end of $E(\sigma_B^S[S_{\max}^S])$, there are at most $|H_1|/k$ shared objects that are poised to be accessed by processes in H_1 , and thus there are at most $2|H_1|/k$ processes removed in the construction of H_2 from H_1 .

Then let H_3 be a modification of H_2 such that each remaining group of H_2 with at least $k/4$ processes is shrunk to contain only $k/4$ processes, and all other groups are removed. Since at most $2|H_1|/k$ processes were removed in the construction of H_2 from H_1 , and $k \geq w \in \Omega(\log n)$, at least half of the processes remain, and so it is easy to see that at least a quarter of the groups in H_2 remain with at least $k/4$ processes. So $|H_3| \geq |H_1|/16 > |H|/32$.

Finally, let $\{X, X'\}$ be a partition of H_3 such that each group in H_3 is in X if and only if no process in the group is poised to perform a read operation at the end of $E(\sigma_B^S[S_{max}^S])$. Then there are two cases: either $|X| \leq |X'|$, or $|X| > |X'|$.

A.7 Read High Contention Case: $|X| \leq |X'|$

Let S_Y be the set of every process in X' that is poised to perform a read operation at the end of $E(\sigma_B^S[S_{max}^S])$.

LEMMA 16. $|S_Y| > |H|/16k$.

PROOF. Recall that X' is composed of groups of exactly $k/4$ processes, and each group contains at least one process that is poised to perform a read operation at the end of $E(\sigma_B^S[S_{max}^S])$. So $|S_Y| \geq 4|X'|/k$. Furthermore, since $|X| \leq |X'|$, $|X'| \geq |H_3|/2 > |H|/64$. Thus $|S_Y| > |H|/16k$. \square

Next, let $S_G = S_Y \cup F(\sigma_B^S[S_{max}^S])$. We now construct a new array $\sigma_A^G[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_G$, then $\sigma_A^G[S] = \perp$; otherwise $\sigma_A^G[S] = \sigma_B^S[S]$.

LEMMA 17. *This new array $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$.*

PROOF. By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. By construction, $\sigma_A^G[0..2^n - 1]$ is simply a modification of $\sigma_B^S[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains any process in $S_{max}^S \setminus S_G$ has had $\sigma_A^G[S]$ set to \perp , where $F(\sigma_B^S[S_{max}^S]) \subseteq S_G \subseteq S_{max}^S$. It suffices to observe that every invariant still holds with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$, and thus $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$. \square

Next, for each set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, let C_S^G be the configuration at the end of $E(\sigma_A^G[S])$, and let σ_S^G be the schedule consisting of exactly one non-crash step by each process in $S \setminus F(\sigma_A^G[S_G])$ in order from the process with the smallest ID to the process with the largest ID. Note that σ_S^G is a finite schedule since there are only a finite number of processes in the system. Also note that by Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$, so $\sigma_A^G[S] \neq \perp$ if and only if $F(\sigma_A^G[S_G]) \subseteq S \subseteq S_G$.

LEMMA 18. *For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$:*

- (G1) *For each shared object $R \in \mathcal{R}$, R can only be accessed by read operations during $E(C_S^G, \sigma_S^G)$.*
- (G2) *For each shared object $R \in \mathcal{R}$ that is accessed during $E(C_S^G, \sigma_S^G)$, the owner of R is not in $S_G \setminus F(\sigma_A^G[S_G])$.*
- (G3) *For each shared object $R \in \mathcal{R}$ that is accessed during $E(C_S^G, \sigma_S^G)$, $last_R(\sigma_A^G[S_G]) \notin S_G \setminus F(\sigma_A^G[S_G])$.*
- (G4) *For each shared object $R \in \mathcal{R}$ that is accessed during $E(C_S^G, \sigma_S^G)$, $val_R(\sigma_A^G[S_G]) = val_R(\sigma_A^G[S])$.*
- (G5) *For each shared object $R \in \mathcal{R}$ that is accessed during $E(C_S^G, \sigma_S^G)$, $val_R(C_{S_G}^G, \sigma_{S_G}^G) = val_R(C_S^G, \sigma_S^G)$.*
- (G6) *For each process $p \in S$, $state_p(C_{S_G}^G, \sigma_{S_G}^G) = state_p(C_S^G, \sigma_S^G)$.*
- (G7) *Each process in $S \setminus F(\sigma_A^G[S_G])$ incurs exactly one RMR during $E(C_S^G, \sigma_S^G)$.*
- (G8) *For each process $p \in S \setminus F(\sigma_A^G[S_G])$, p has not left the critical section during $E(\sigma_A^G[S] \circ \sigma_S^G)$.*
- (G9) *$F(\sigma_A^G[S]) = F(\sigma_A^G[S] \circ \sigma_S^G)$.*

PROOF. Let $S \subseteq \mathcal{P}$ be any set of processes such that $\sigma_A^G[S] \neq \perp$. By Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$. So by Invariant (I2), for every set $S' \subseteq \mathcal{P}$, $\sigma_A^G[S'] \neq \perp$ if and only if $F(\sigma_A^G[S_G]) \subseteq S' \subseteq S_G$. Thus $F(\sigma_A^G[S_G]) \subseteq S \subseteq S_G$.

By [Lemma 10](#), $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_B^S[0..2^n - 1]) = S_{\max}^S$. By [Lemma 17](#), $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^G[0..2^n - 1]) = S_G$. By construction, $\sigma_A^G[S_G] = \sigma_B^S[S_G]$. Thus by [Invariants \(I3\)](#) and [\(I4\)](#), for every process $p \in S$, $\text{state}_p(\sigma_A^G[S]) = \text{state}_p(\sigma_A^G[S_G]) = \text{state}_p(\sigma_B^S[S_G]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$ and $F(\sigma_A^G[S]) = F(\sigma_A^G[S_G]) = F(\sigma_B^S[S_G]) = F(\sigma_B^S[S_{\max}^S])$.

Suppose, for contradiction, that some shared object R is accessed by a non-read operation during $E(C_S^G, \sigma_S^G)$. Let $p \in \mathcal{P}$ be a process that performs a non-read operation on R during $E(C_S^G, \sigma_S^G)$. By definition, σ_S^G consists of exactly one non-crash step by each process in $S \setminus F(\sigma_A^G[S_G])$. So p is in $S \setminus F(\sigma_A^G[S_G])$. Then, since $S \subseteq S_G$ and $F(\sigma_A^G[S_G]) = F(\sigma_B^S[S_{\max}^S])$, p is in $S_G \setminus F(\sigma_B^S[S_{\max}^S])$. By definition, $S_G = S_Y \cup F(\sigma_B^S[S_{\max}^S])$. Thus p is in S_Y . By definition, S_Y is the set of every process in X' that is poised to perform a read operation at the end of $E(\sigma_B^S[S_{\max}^S])$. So p is poised to perform a read operation at the end of $E(\sigma_B^S[S_{\max}^S])$. Next, since p is in S , $\text{state}_p(\sigma_A^G[S]) = \text{state}_p(\sigma_B^S[S_{\max}^S])$. Thus p is also poised to perform a read operation in the configuration C_S^G at the end of $E(\sigma_A^G[S])$. Finally, since σ_S^G consists of exactly one step of p , p only performs a read operation during $E(C_S^G, \sigma_S^G)$ – contradicting that p performs a non-read operation on R during $E(C_S^G, \sigma_S^G)$. Thus for each shared object $R \in \mathcal{R}$, R can only be accessed by read operations during $E(C_S^G, \sigma_S^G)$ [\(G1\)](#).

Let $R \in \mathcal{R}$ be a shared object that is accessed during $E(C_S^G, \sigma_S^G)$. By definition, σ_S^G consists of exactly one non-crash step by each process in $S \setminus F(\sigma_A^G[S_G])$. Thus there exists a process $q \in S \setminus F(\sigma_A^G[S_G])$ that is poised to access R in the configuration C_S^G at the end of $E(\sigma_A^G[S])$. Since $S \subseteq S_G$ and $F(\sigma_A^G[S_G]) = F(\sigma_B^S[S_{\max}^S])$, q is in $S_G \setminus F(\sigma_B^S[S_{\max}^S])$. By definition, $S_G = S_Y \cup F(\sigma_B^S[S_{\max}^S])$. Thus q is in S_Y . By definition, $S_Y \subseteq X' \subseteq H_3 \subseteq H_2 \subseteq H_1$. So q is in H_1 . Furthermore, since q is in S , $\text{state}_q(\sigma_A^G[S]) = \text{state}_q(\sigma_B^S[S_{\max}^S])$. Thus $q \in H_1$ is also poised to access R at the end of $E(\sigma_B^S[S_{\max}^S])$.

Now recall that H_2 is a modification of H_1 such that for each process $p \in H_1$, p is in H_2 if and only if both of the following are true:

- No process in H_1 is poised to access a shared object owned by p at the end of $E(\sigma_B^S[S_{\max}^S])$.
- No process in H_1 is poised to access a shared object R' at the end of $E(\sigma_B^S[S_{\max}^S])$ such that $\text{last}_{R'}(\sigma_B^S[S_{\max}^S]) = p$.

Furthermore, recall that $S_G \setminus F(\sigma_A^G[S_G]) \subseteq S_Y \subseteq H_2$. Consequently:

- The owner of R is not in $S_G \setminus F(\sigma_A^G[S_G])$ [\(G2\)](#).
- $\text{last}_R(\sigma_A^G[S_G]) \notin S_G \setminus F(\sigma_A^G[S_G])$ [\(G3\)](#).

By [Lemma 17](#), $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^G[0..2^n - 1]) = S_G$. Thus by [Invariant \(I5\)](#), since $\text{last}_R(\sigma_A^G[S_G]) \notin S_G \setminus F(\sigma_A^G[S_G])$ and $F(\sigma_A^G[S_G]) \subseteq S \subseteq S_G$, $\text{val}_R(\sigma_A^G[S_G]) = \text{val}_R(\sigma_A^G[S])$ [\(G4\)](#). Furthermore, since we have already proven that [\(G1\)](#) holds, R is only accessed by read operations during both $E(C_S^G, \sigma_S^G)$ and $E(C_{S_G}^G, \sigma_{S_G}^G)$. Thus, since read operations cannot change the state of a shared object, $\text{val}_R(C_{S_G}^G, \sigma_{S_G}^G) = \text{val}_R(\sigma_A^G[S_G]) = \text{val}_R(\sigma_A^G[S]) = \text{val}_R(C_S^G, \sigma_S^G)$. [\(G5\)](#).

Next, recall that for each process $p \in S$, $\text{state}_p(\sigma_A^G[S]) = \text{state}_p(\sigma_A^G[S_G])$. By definition, each of σ_S^G and $\sigma_{S_G}^G$ contains exactly one non-crash step of each process $S \setminus F(\sigma_A^G[S_G])$, and contains no steps of each process in $F(\sigma_A^G[S_G])$. Furthermore, we have already proven that [\(G1\)](#) and [\(G4\)](#). Thus for each process $p \in S$:

- If $p \in F(\sigma_A^G[S_G])$, then p takes no steps in both $E(C_{S_G}^G, \sigma_{S_G}^G)$ and $E(C_S^G, \sigma_S^G)$. Thus since $\text{state}_p(\sigma_A^G[S]) = \text{state}_p(\sigma_A^G[S_G])$, $\text{state}_p(C_{S_G}^G, \sigma_{S_G}^G) = \text{state}_p(C_S^G, \sigma_S^G)$.
- Otherwise, there is a shared object R' such that (i) $\text{val}_{R'}(\sigma_A^G[S_G]) = \text{val}_{R'}(\sigma_A^G[S])$, (ii) R' is only accessed by read operations during both $E(C_{S_G}^G, \sigma_{S_G}^G)$ and $E(C_S^G, \sigma_S^G)$, and (iii) p performs exactly one read operation on R' during both $E(C_{S_G}^G, \sigma_{S_G}^G)$ and $E(C_S^G, \sigma_S^G)$. Thus,

since $state_p(\sigma_A^G[S]) = state_p(\sigma_A^G[S_G])$ and read operations cannot change the state of R' , $state_p(C_{S_G}^G, \sigma_{S_G}^G) = state_p(C_S^G, \sigma_S^G)$ (G6).

Next, since $\sigma_A^G[S] \neq \perp$, by construction, $\sigma_{old}[S] \neq \perp$. By Lemma 8 (S3), for each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p incurs an RMR at the end of $E(\sigma_A^S[S] \circ p)$. By construction and Lemma 9 (Invariant (I4)), $\sigma_A^G[S] = \sigma_B^S[S] = \sigma_A^S[S]$, and $F(\sigma_A^G[S_G]) = F(\sigma_A^S[S_G]) = F(\sigma_A^S[S_{max}^{old}]) = F(\sigma_{old}[S_{max}^{old}])$. Thus for each process $p \in S \setminus F(\sigma_A^G[S_G])$, p incurs an RMR at the end of $E(\sigma_A^G[S] \circ p)$. Now recall that by definition, σ_S^G consists of exactly one non-crash step by each process in $S \setminus F(\sigma_A^G[S_G])$. Consequently, observe that each process in $S \setminus F(\sigma_A^G[S_G])$ must incur exactly one RMR during $E(C_S^G, \sigma_S^G)$ (G7).

By Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I7), each process that is not in $F(\sigma_A^G[S])$ does not enter the critical section during $E(\sigma_A^G[S])$. Recall that σ_S^G consists of exactly one non-crash step of each process in $S \setminus F(\sigma_A^G[S_G])$ and no other steps. Thus, with only one step, although a process could enter the critical section during $E(C_S^G, \sigma_S^G)$, it cannot have taken any steps within the critical section. Thus by (A2), no process can leave the critical section during $E(C_S^G, \sigma_S^G)$. So for each process $p \in S \setminus F(\sigma_A^G[S_G])$, p has not left the critical section during $E(\sigma_A^G[S] \circ \sigma_S^G)$ (G8). Therefore, since no process in $S \setminus F(\sigma_A^G[S_G])$ has left the critical section during $E(\sigma_A^G[S] \circ \sigma_S^G)$, no process in $S \setminus F(\sigma_A^G[S_G])$ has completed its super-passage during $E(\sigma_A^G[S] \circ \sigma_S^G)$. Thus $F(\sigma_A^G[S]) = F(\sigma_A^G[S] \circ \sigma_S^G)$ (G9). \square

We now construct a new array $\sigma_B^G[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $\sigma_A^G[S] = \perp$ then $\sigma_B^G[S] = \perp$; otherwise $\sigma_B^G[S] = \sigma_A^G[S] \circ \sigma_S^G$.

LEMMA 19. *Except for Invariant (I7), this new array $\sigma_B^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_B^G[0..2^n - 1]) = S_G$.*

PROOF. For every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, then by construction, $\sigma_B^G[S] = \sigma_A^G[S] \circ \sigma_S^G$. By Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I1), $P(\sigma_A^G[S]) \subseteq S$. By the definition of σ_S^G , σ_S^G contains only steps of processes in $S \setminus F(\sigma_A^G[S_G])$. Thus $P(\sigma_B^G[S]) \subseteq S$ (Invariant (I1)).

By Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$. So by Invariant (I2), for every set $S \subseteq \mathcal{P}$, $\sigma_A^G[S] \neq \perp$ if and only if $F(\sigma_A^G[S_G]) \subseteq S \subseteq S_G$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_B^G[S] = \perp$ if and only if $\sigma_A^G[S] = \perp$. Furthermore, by Lemma 18 (G9), $F(\sigma_B^G[S_G]) = F(\sigma_A^G[S_G])$. Thus for every set $S \subseteq \mathcal{P}$, $\sigma_B^G[S] \neq \perp$ if and only if $F(\sigma_B^G[S_G]) \subseteq S \subseteq S_G$ (Invariant (I2)).

By Lemma 18 (G6), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, and every process $p \in S$, $state_p(\sigma_B^G[S_G]) = state_p(C_{S_G}^G, \sigma_{S_G}^G) state_p(C_S^G, \sigma_S^G) = state_p(\sigma_B^G[S])$. By construction, for every set $S \subseteq \mathcal{P}$, $\sigma_B^G[S] = \perp$ if and only if $\sigma_A^G[S] = \perp$. Furthermore, we have already proven that for every set $S \subseteq \mathcal{P}$, $\sigma_B^G[S] \neq \perp$ if and only if $F(\sigma_B^G[S_G]) \subseteq S \subseteq S_G$. Thus for every process $p \in S_G$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_B^G[S] \neq \perp$, then $state_p(\sigma_B^G[S]) = state_p(\sigma_B^G[S_G])$ (Invariant (I3)).

Since we have already proven that Invariants (I1), (I2), and (I3) hold for $\sigma_B^G[0..2^n - 1]$, it immediately follows that Invariant (I4) also holds.

Next, by Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^G[0..2^n - 1]) = S_G$. So by Invariant (I5), for every shared object $R \in \mathcal{R}$, there is a value y_R such that for every set $S \subseteq \mathcal{P}$, if $\sigma_A^G[S] \neq \perp$, then:

$$val_R(\sigma_A^G[S]) = \begin{cases} val_R(\sigma_A^G[S_G]) & \text{if } last_R(\sigma_A^G[S_G]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Recall that by construction, for every set $S \subseteq \mathcal{P}$, $\sigma_A^G[S] \neq \perp$ if and only if $\sigma_B^G[S] \neq \perp$. Furthermore, by [Lemma 18 \(G1\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, shared objects can only be accessed by read operations during $E(C_S^G, \sigma_S^G)$. Thus, since read operations cannot change the state of a shared object, observe that for every shared object $R \in \mathcal{R}$ and for every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, then:

$$\text{val}_R(\sigma_B^G[S]) = \begin{cases} \text{val}_R(\sigma_B^G[S_G]) & \text{if } \text{last}_R(\sigma_A^G[S_G]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Therefore, for each shared object $R \in \mathcal{R}$:

- If R is accessed during $E(C_{S_G}^G, \sigma_{S_G}^G)$, then $\text{last}_R(\sigma_A^G[S_G]) \notin S_G \setminus F(\sigma_A^G[S_G])$ by [Lemma 18 \(G3\)](#). Recall that $\sigma_B^G[S] \neq \perp$ if and only if $F(\sigma_B^G[S_G]) = F(\sigma_A^G[S_G]) \subseteq S \subseteq S_G$. So for every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, then either $\text{last}_R(\sigma_A^G[S_G])$ is in both S and S_G or both S and S_G do not contain $\text{last}_R(\sigma_A^G[S_G])$. Thus for every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, $\text{val}_R(\sigma_B^G[S]) = \text{val}_R(\sigma_B^G[S_G])$.
- If R is not accessed during $E(C_{S_G}^G, \sigma_{S_G}^G)$, then it is clear that $\text{last}_R(\sigma_B^G[S_G]) = \text{last}_R(\sigma_A^G[S_G])$. Thus for every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, then:

$$\text{val}_R(\sigma_B^G[S]) = \begin{cases} \text{val}_R(\sigma_B^G[S_G]) & \text{if } \text{last}_R(\sigma_B^G[S_G]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Consequently, Invariant [\(I5\)](#) holds for $\sigma_B^G[0..2^n - 1]$.

By [Lemma 17](#), $\sigma_A^G[0..2^n - 1]$ is $(i-1)$ -compliant, so Invariant [\(I6\)](#) holds for $\sigma_A^G[0..2^n - 1]$. By definition, for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, $\sigma_B^G[S] = \sigma_A^G[S] \circ \sigma_S^G$ and σ_S^G contains no crash steps. Thus Invariant [\(I6\)](#) also holds for $\sigma_B^G[0..2^n - 1]$.

By [Lemma 17](#), $\sigma_A^G[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_A^G[0..2^n - 1]) = S_G$. So by Invariant [\(I8\)](#), for every process $p \in S_G \setminus F(\sigma_A^G[S_G])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $\sigma_A^G[S] \neq \perp$, R can only be accessed by p during $E(\sigma_A^G[S])$. By [Lemma 18 \(G2\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, for each shared object $R \in \mathcal{R}$ that is accessed during $E(C_S^G, \sigma_S^G)$, the owner of R is not in $S_G \setminus F(\sigma_A^G[S_G])$. Thus for every process $p \in S_G \setminus F(\sigma_A^G[S_G])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $\sigma_A^G[S] \neq \perp$, R is not accessed during $E(C_S^G, \sigma_S^G)$. Now recall that (i) $F(\sigma_B^G[S_G]) = F(\sigma_A^G[S_G])$ by [Lemma 18 \(G9\)](#), (ii) $\sigma_B^G[S] \neq \perp$ if and only if $\sigma_A^G[S] \neq \perp$, and (iii) if $\sigma_B^G[S] \neq \perp$ then $\sigma_B^G[S] = \sigma_A^G[S] \circ \sigma_S^G$. Consequently, for every process $p \in S_G \setminus F(\sigma_B^G[S_G])$, every shared object $R \in \mathcal{R}$ owned by p , and every set $S \subseteq \mathcal{P}$ with $\sigma_B^G[S] \neq \perp$, R can only be accessed by p during $E(\sigma_B^G[S])$ (Invariant [\(I8\)](#)).

For every process $p \in S_G \setminus F(\sigma_A^G[S_G])$, let R_p be the shared object that p is poised to access at the end of $E(\sigma_A^G[S_G])$. By [Lemma 17](#), $\sigma_A^G[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_A^G[0..2^n - 1]) = S_G$. So by Invariant [\(I3\)](#), for every process $p \in S_G$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^G[S] \neq \perp$, then $\text{state}_p(\sigma_A^G[S]) = \text{state}_p(\sigma_A^G[S_G])$, and so p is also poised to access R_p at the end of $E(\sigma_A^G[S])$. Furthermore, by Invariant [\(I9\)](#), in the CC model, for every process $p \in S_G \setminus F(\sigma_A^G[S_G])$, there is a set \mathcal{R}_p of shared objects such that for every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^G[S] \neq \perp$, then the set of shared objects that p has valid cache copies of at the end of $E(\sigma_A^G[S])$ is exactly \mathcal{R}_p . By [Lemma 18 \(G1\)](#), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, shared objects can only be accessed by read operations during $E(C_S^G, \sigma_S^G)$. Now recall that (i) $F(\sigma_B^G[S_G]) = F(\sigma_A^G[S_G])$ by [Lemma 18 \(G9\)](#), (ii) for every set $S \subseteq \mathcal{P}$, $\sigma_A^G[S] \neq \perp$ if and only if $\sigma_B^G[S] \neq \perp$, and (iii) for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, σ_S^G consists of exactly one non-crash step by each process in $S \setminus F(\sigma_A^G[S_G])$ and no other steps. Consequently, observe that for every process $p \in S_G \setminus F(\sigma_B^G[S_G])$, for every set $S \subseteq \mathcal{P}$

that contains p , if $\sigma_B^G[S] \neq \perp$, then the set of shared objects that p has valid cache copies of at the end of $E(\sigma_A^G[S])$ is exactly $\mathcal{R}_p \cup \{R_p\}$ (Invariant (I9)).

By Lemma 17, $\sigma_A^G[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I10), for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_A^G[S])$, if $\sigma_A^G[S] \neq \perp$, then p incurs at least $i - 1$ RMRs during $E(\sigma_A^G[S])$. By Lemma 18 (L8), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^G[S] \neq \perp$, each process in $S \setminus F(\sigma_A^G[S_G])$ incurs exactly one RMR during $E(C_S^G, \sigma_S^G)$. By construction, for every set $S \subseteq \mathcal{P}$, if $\sigma_B^G[S] \neq \perp$, then $\sigma_B^G[S] = \sigma_A^G[S] \circ \sigma_S^G$, i.e., every process in $S \setminus F(\sigma_A^G[S_G])$ incurs exactly one more RMR during $E(\sigma_B^G[S])$ than during $E(\sigma_A^G[S])$. By Lemma 18 (G9), $F(\sigma_A^G[S_G]) = F(\sigma_B^G[S_G])$. Since we have already proven that Invariant (I4) holds for $\sigma_B^G[0..2^n - 1]$, $F(\sigma_B^G[S_G]) = F(\sigma_B^G[S])$ for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^G[S] \neq \perp$. Thus for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_B^G[S])$, if $\sigma_B^G[S] \neq \perp$, then p incurs at least i RMRs during $E(\sigma_A^G[S])$ (Invariant (I10)). \square

We now construct another array $\sigma_C^G[0..2^n - 1]$ with the goal of satisfying Invariant (I7) as follows. If no process is within the critical section at the end of $E(\sigma_B^G[S_G])$, we simply construct $\sigma_C^G[0..2^n - 1]$ such that $\sigma_C^G[0..2^n - 1] = \sigma_B^G[0..2^n - 1]$. Furthermore, we define $S_{max}^G = S_G$.

Otherwise, to avoid violating mutual exclusion, there must be exactly one process $p \in S_G \setminus F(\sigma_B^G[S_G])$ such that at the end of $E(\sigma_B^G[S_G])$, p is within the critical section. Then note that by Lemma 19 and Invariant (I3), for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^G[S] \neq \perp$, if $p \in S$ then p is also within the critical section at the end of $E(\sigma_B^G[S])$; otherwise no process is within the critical section at the end of $E(\sigma_B^G[S])$. Thus we construct $\sigma_C^G[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^G[S] = \perp$; otherwise $\sigma_C^G[S] = \sigma_B^G[S]$. Furthermore, we define $S_{max}^G = S_G \setminus \{p\}$.

LEMMA 20. *This new array $\sigma_C^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_C^G[0..2^n - 1]) = S_{max}^G$.*

PROOF. By Lemma 19, except for Invariant (I7), $\sigma_B^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_B^G[0..2^n - 1]) = S_G$.

If no process is within the critical section at the end of $E(\sigma_B^G[S_G])$, then $\sigma_C^G[0..2^n - 1] = \sigma_B^G[0..2^n - 1]$. Thus by Lemma 18 (G8) and Lemma 19, Invariant (I7) also holds for $\sigma_C^G[0..2^n - 1]$, and so it follows that $\sigma_C^G[0..2^n - 1] = \sigma_B^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_C^G[0..2^n - 1]) = S_{max}^G = S_G$.

Otherwise, there is exactly one process $p \in S_G \setminus F(\sigma_B^G[S_G])$ such that p is within the critical section at the end of $E(\sigma_B^G[S_G])$, and for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^G[S] = \perp$; otherwise $\sigma_C^G[S] = \sigma_B^G[S]$. By Lemma 19, except for Invariant (I7), $\sigma_B^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_B^G[0..2^n - 1]) = S_G$. Thus observe that by the construction of $\sigma_C^G[0..2^n - 1]$, Invariants (I1), (I2), (I3), (I4), (I5), (I6), (I8), (I9), (I10) must all also hold for $\sigma_C^G[0..2^n - 1]$ with $S_{max}(\sigma_C^G[0..2^n - 1]) = S_{max}^G = S_G \setminus \{p\}$.

By the construction of $\sigma_C^G[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $p \in S$, then $\sigma_C^G[S] = \perp$. Thus for every set $S \subseteq \mathcal{P}$, if $\sigma_C^G[S] \neq \perp$, then $p \notin S$. Consequently, for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^G[S] \neq \perp$, no process is within the critical section at the end of $E(\sigma_C^G[S]) = E(\sigma_B^G[S])$. Therefore by Lemma 18 (L9), Invariant (I7) holds for $\sigma_C^G[0..2^n - 1]$, and thus $\sigma_C^G[0..2^n - 1]$ is i -compliant with $S_{max}(\sigma_C^G[0..2^n - 1]) = S_{max}^G$. \square

Finally, we terminate this i -th iteration by setting $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^G[0..2^n - 1]$.

A.8 Non-Read High Contention Case: $|X| > |X'|$

Let m be the number of groups in X . Then since X only contains groups with exactly $k/4$ processes, $m = 4|X|/k$.

We arbitrarily order the shared objects in \mathcal{R} , then order the groups in X by the order in which the shared object that each group is poised to access appears in the ordering of \mathcal{R} , with ties broken

arbitrarily. Thus note that all groups that are poised to access the same shared object are consecutive in the ordering of groups. Then for every integer $j \in \{1, 2, \dots, m\}$, let X_j be the j -th group in the ordering and R_j be the shared object that every process in X_j is poised to access at the end of $E(\sigma_B^S[S_{max}^S])$. Note that since X only contains processes that are in groups, $\{X_1, \dots, X_m\}$ is a partition of X .

Finally, let $S_H = F(\sigma_B^S[S_{max}^S]) \cup X$. We now construct a new array $\sigma_A^H[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_H$, then $\sigma_A^H[S] = \perp$; otherwise $\sigma_A^H[S] = \sigma_B^S[S]$.

LEMMA 21. *This new array $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$. Furthermore, for every integer $j \in \{1, 2, \dots, m\}$:*

- For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_A^H[S])$.
- The owner of R_j is not in $S_H \setminus F(\sigma_A^H[S_H])$.
- For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $val_{R_j}(\sigma_A^H[S]) = val_{R_j}(\sigma_A^H[S_H])$.

PROOF. By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. By construction, $\sigma_A^H[0..2^n - 1]$ is simply a modification of $\sigma_B^S[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains any process in $S_{max}^S \setminus S_H$ has had $\sigma_A^H[S]$ set to \perp , where $F(\sigma_B^S[S_{max}^S]) \subseteq S_H \subseteq S_{max}^S$. It suffices to observe that every invariant still holds with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$, and thus $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$.

Now let j be an integer in $\{1, 2, \dots, m\}$. By definition, R_j is the shared object that every process in X_j is poised to access at the end of $E(\sigma_B^S[S_{max}^S])$. By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. So by Invariant (I3), for every process $p \in S_{max}^S$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_B^S[S] \neq \perp$, then $state_p(\sigma_B^S[S]) = state_p(\sigma_B^S[S_{max}^S])$. Thus for every process $p \in X_j$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_B^S[S] \neq \perp$, then p is also poised to access R_j at the end of $E(\sigma_B^S[S])$.

By the construction of $\sigma_A^H[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $\sigma_A^H[S] = \sigma_B^S[S]$. So for every process $p \in X_j$ and every set $S \subseteq \mathcal{P}$ that contains p , if $\sigma_A^H[S] \neq \perp$, then p is also poised to access R_j at the end of $E(\sigma_A^H[S])$. Thus for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_A^H[S])$.

Now recall that $S_H = F(\sigma_B^S[S_{max}^S]) \cup X$, where $X \subseteq H_2$. By construction, for each process $p \in H_1$, p is in H_2 if and only if both of the following are true:

- No process in H_1 is poised to access a shared object owned by p at the end of $E(\sigma_B^S[S_{max}^S])$.
- No process in H_1 is poised to access a shared object R at the end of $E(\sigma_B^S[S_{max}^S])$ such that $last_R(\sigma_B^S[S_{max}^S]) = p$.

We have already shown that $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$, and that for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_A^H[S])$. So by Invariant (I2), every process in $X_j \cap S_H = X_j$ is poised to access R_j at the end of $E(\sigma_A^H[S_H])$. Thus:

- The owner of R_j is not in $S_H \setminus F(\sigma_A^H[S_H])$.
- $last_{R_j}(\sigma_B^S[S_{max}^S])$ is not in $S_H \setminus F(\sigma_A^H[S_H])$.

By Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. So by Invariant (I5), for every shared object $R \in \mathcal{R}$, there is a value y_R such that for every set $S \subseteq \mathcal{P}$, if $\sigma_B^S[S] \neq \perp$, then:

$$val_R(\sigma_B^S[S]) = \begin{cases} val_R(\sigma_B^S[S_{max}^S]) & \text{if } last_R(\sigma_B^S[S_{max}^S]) \in S \\ y_R & \text{otherwise} \end{cases}$$

We have just shown that $last_{R_j}(\sigma_B^S[S_{max}^S])$ is not in $S_H \setminus F(\sigma_A^H[S_H])$. So either $last_{R_j}(\sigma_B^S[S_{max}^S]) \in F(\sigma_A^H[S_H])$ or $last_{R_j}(\sigma_B^S[S_{max}^S]) \notin S_H$.

By the construction of $\sigma_A^H[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $\sigma_A^H[S] \neq \perp$, then $\sigma_A^H[S] = \sigma_B^S[S]$. Thus if $last_{R_j}(\sigma_B^S[S_{max}^S])$ is in $F(\sigma_A^H[S_H])$, then for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $val_{R_j}(\sigma_A^H[S]) = val_{R_j}(\sigma_B^S[S_{max}^S])$.

Otherwise $last_{R_j}(\sigma_B^S[S_{max}^S])$ is not in S_H . Since we have already proven that $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$, by Invariant (I2), for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $S \subseteq S_H$, and so $last_{R_j}(\sigma_B^S[S_{max}^S])$ is not in S . Thus for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $val_R(\sigma_A^H[S]) = y_R$.

So in both cases, for every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $val_{R_j}(\sigma_A^H[S]) = val_{R_j}(\sigma_A^H[S_H])$. \square

Now for every set $S \subseteq X$, let σ_S^* be the schedule that is composed of exactly one non-crash step by each process in S , in order from the smallest process id to largest process id. Then for every value $y \in Y$, we define a function $f_y : 2^X \rightarrow Y$ such that for every set $S \subseteq X$:

- Let $j \in \{1, 2, \dots, m\}$ be an integer such that $S \subseteq X_j$. If j does not exist, then $f_y(S)$ outputs an arbitrary value in Y .
- Otherwise, note that by Lemma 21, every process in S is poised to access R_j at the end of $E(\sigma_A^H[S])$. If either $j = 1$ or $R_{j-1} \neq R_j$, then $f_y(S) = val_{R_j}(\sigma_A^H[S_H] \circ \sigma_S^*)$.
- Otherwise, let $\hat{\sigma}$ be any schedule composed of steps of processes in $X \setminus X_j$ such that $y = val_{R_j}(\sigma_A^H[S_H] \circ \hat{\sigma})$. If $\hat{\sigma}$ does not exist, then $f_y(S)$ outputs an arbitrary value in Y .
- Otherwise, note that regardless of the choice of $\hat{\sigma}$, since $\hat{\sigma}$ does not contain any steps of S , the state of every process in S is the same in $E(\sigma_A^H[S])$ as in $E(\sigma_A^H[S] \circ \hat{\sigma})$. So every process in S is still poised to access R_j at the end of $E(\sigma_A^H[S] \circ \hat{\sigma})$. Then $f_y(S) = val_{R_j}(\sigma_A^H[S_H] \circ \hat{\sigma} \circ \sigma_S^*)$.

Next, let $\ell = w \geq 0$, $\delta = 2w \geq 1$, and y_0 be any value in the domain Y of values that the shared objects can store.

LEMMA 22. *There exists $y_j \in Y$ and non-empty sets $A_j \subseteq V_j \subseteq X_j$ for every integer $j \in \{1, 2, \dots, m\}$, such that $f_{y_{j-1}}(A_j) = y_j$ for all j , and for any given set $D \subseteq X$ of size $|D| \leq \delta \cup_j V_j$, there exists a subset of indices $I^D \subseteq \{1, \dots, m\}$ of size $|I^D| \geq m/2$, and for each $j \in I^D$ there exist $B_j^D \subseteq V_j$ and $z_j^D \in X_j \setminus (V_j \cup D)$ such that $f_{y_{j-1}}(B_j^D \cup \{z_j^D\}) = y_j$.*

PROOF. Since $\ell = w$, $|Y| \leq 2^\ell$. Furthermore, for every integer $j \in \{1, 2, \dots, m\}$, $|X_j| \geq 108\delta\ell^2$ because each group contains exactly $k/4$ processes and $k = w^d$ for some sufficiently large constant d . So the lemma immediately follows from Lemma 2. \square

For every integer $j \in \{1, 2, \dots, m\}$, let $\alpha[j] = \sigma_{A_j}^*$, i.e., $\alpha[j]$ is the schedule that is composed of exactly one non-crash step by each process in A_j , in order from the smallest process id to largest process id. Then let $\sigma_\alpha^0 = \emptyset$ and σ_α^j be the concatenation of all schedules in $\alpha[1..j]$, i.e., $\sigma_\alpha^j = \alpha[1] \circ \alpha[2] \circ \dots \circ \alpha[j]$. Then let $\sigma_\alpha = \sigma_\alpha^m$.

Similarly, given any set $D \subseteq X$ of size $|D| \leq \delta \cup_j V_j$, for every integer $j \in I^D$, let $\beta^D[j] = \sigma_{B_j^D \cup \{z_j^D\}}^*$, i.e., $\beta^D[j]$ is the schedule that is composed of exactly one non-crash step by each process in $B_j^D \cup \{z_j^D\}$, in order from the smallest process id to largest process id. Then for every set $I' \subseteq I^D$, let $\sigma_{I',D}^0 = \emptyset$ and

$$\sigma_{I',D}^j = \begin{cases} \sigma_{I',D}^{j-1} \circ \beta^D[j] & \text{if } j \in I' \\ \sigma_{I',D}^{j-1} \circ \alpha[j] & \text{otherwise} \end{cases}$$

Then let $\sigma_{I',D} = \sigma_{I',D}^m$.

Next, let S_α be the set of all processes in $\{V_1, V_2, \dots, V_m\}$. So note that S_α also contains every process in σ_α . Furthermore, by [Lemma 22](#), given any set $D \subseteq X$ of size $|D| \leq \delta|\bigcup_j V_j|$, for each $j \in I^D$, $B_j^D \subseteq V_j \subseteq S_\alpha$.

LEMMA 23. *For every set $D \subseteq X$ of size $|D| \leq \delta|\bigcup_j V_j|$, every set $I' \subseteq I^D$, and every integer $j \in \{1, 2, \dots, m\}$:*

- (a) $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = y_j$.
- (b) For every set $I'' \subseteq I^D$ such that both $\sigma_{I',D}$ and $\sigma_{I'',D}$ contain the contiguous subsequence $\beta^D[j]$, $\text{state}_{z_j^D}(\sigma_A^H[S_H] \circ \sigma_{I',D}) = \text{state}_{z_j^D}(\sigma_A^H[S_H] \circ \sigma_{I'',D})$.
- (c) For every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_A^H[S_H]) \cup S_\alpha \subseteq S \subseteq S_H$ and $z_{j'}^D \in S$ for every $j' \in I'$, both $\text{val}_{R_j}(\sigma_A^H[S] \circ \sigma_{I',D}^j) = y_j$ and for every process $p \in S$, $\text{state}_p(\sigma_A^H[S] \circ \sigma_{I',D}^j) = \text{state}_p(\sigma_A^H[S_H] \circ \sigma_{I',D}^j)$.
- (d) For every set $I'' \subseteq I^D$ such that both $\sigma_{I',D}$ and $\sigma_{I'',D}$ contain the contiguous subsequence $\beta^D[j]$, and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_A^H[S_H]) \cup S_\alpha \subseteq S \subseteq S_H$ and $z_{j'}^D \in S$ for every $j' \in I' \cup I''$, $\text{state}_{z_j^D}(\sigma_A^H[S] \circ \sigma_{I',D}) = \text{state}_{z_j^D}(\sigma_A^H[S] \circ \sigma_{I'',D})$.

PROOF. (a): We will prove this by induction. Let $j \in \{1, 2, \dots, m\}$ be an integer. If $j > 1$, assume that $\text{val}_{R_{j-1}}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = y_{j-1}$. Then it suffices to prove that $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = y_j$.

By [Lemma 22](#), observe that by definition, $\sigma_{I',D}^{j-1}$ contains only steps of processes in $X_1 \cup X_2 \cup \dots \cup X_{j-1}$. So the state of each process in X_j at the end of $E(\sigma_A^H[S_H])$ is the same as at the end of $E(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$.

There are two cases: either $j \neq 1$ and $R_{j-1} = R_j$, or not.

Case 1. $j \neq 1$ and $R_{j-1} = R_j$.

Then by assumption, $\text{val}_{R_{j-1}}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = y_{j-1}$. So since $R_{j-1} = R_j$, $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = y_{j-1}$. Thus $\sigma_{I',D}^{j-1}$ is a schedule composed of steps of processes in $X \setminus X_j$ such that $y_{j-1} = \text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$. So by definition, $f_{y_{j-1}}(A_j) = \text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1} \circ \alpha[j])$ and $f_{y_{j-1}}(B_j^D \cup \{z_j^D\}) = \text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1} \circ \beta^D[j])$.

By [Lemma 22](#), since $|D| \leq \delta|\bigcup_j V_j|$ and $I' \subseteq I^D$, both $f_{y_{j-1}}(A_j)$ and $f_{y_{j-1}}(B_j^D \cup \{z_j^D\})$ are equal to y_j . Thus regardless of whether $j \in I'$, $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = y_j$.

Case 2. $j = 1$ or $R_{j-1} \neq R_j$.

Then by the ordering of the shared objects R_1, R_2, \dots, R_m , there is no integer $j' < j$ in $\{1, 2, \dots, m\}$ such that $R_{j'} = R_j$. Thus by [Lemma 21](#), for every integer $j' < j$, every process in $X_{j'}$ is not poised to access R_j at the end of $E(\sigma_A^H[S_H])$.

Next, by [Lemma 22](#), observe that by definition, $\sigma_{I',D}^{j-1}$ is a schedule composed of at most one step by each process in $X_1 \cup X_2 \cup \dots \cup X_{j-1}$. So every access of R_j during $E(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$ occurred during $E(\sigma_A^H[S_H])$. Thus $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = \text{val}_{R_j}(\sigma_A^H[S_H])$.

Now recall that the state of each process in X_j at the end of $E(\sigma_A^H[S_H])$ is the same as at the end of $E(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$. Consequently, $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j)$ equals either $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \alpha[j])$ or $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \beta^D[j])$.

Finally, since $j = 1$ or $R_{j-1} \neq R_j$, $f_{y_{j-1}}(A_j) = \text{val}_{R_j}(\sigma_A^H[S_H] \circ \alpha[j])$ and $f_{y_{j-1}}(B_j^D \cup \{z_j^D\}) = \text{val}_{R_j}(\sigma_A^H[S_H] \circ \beta^D[j])$. By [Lemma 22](#), since $|D| \leq \delta|\bigcup_j V_j|$ and $I' \subseteq I^D$, both $f_{y_{j-1}}(A_j)$ and $f_{y_{j-1}}(B_j^D \cup \{z_j^D\})$ are equal to y_j . Thus regardless of whether $j \in I'$, $\text{val}_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = y_j$.

(b): Let $C_{I'}$ and $C_{I''}$ be the configurations at the end of $E(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$ and $E(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1})$ respectively. We first show that:

- Every process in X_j has the same state in $C_{I'}$ as in $C_{I''}$.
- Every process in X_j is poised to access R_j in both configurations.
- R_j has the same state in $C_{I'}$ as in $C_{I''}$.

By [Lemma 22](#), observe that by definition, both $\sigma_{I',D}^{j-1}$ and $\sigma_{I'',D}^{j-1}$ contain only steps of processes in $X_1 \cup X_2 \cup \dots \cup X_{j-1}$. So for every process $p \in X_j$, $state_p(\sigma_A^H[S_H]) = state_p(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = state_p(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1})$. Thus by [Lemma 21](#), for every process $p \in X_j$, p is poised to access R_j at the end of both $E(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1})$ and $E(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1})$. Therefore by the definitions of $C_{I'}$ and $C_{I''}$:

- Every process in X_j has the same state in $C_{I'}$ as in $C_{I''}$.
- Every process in X_j is poised to access R_j in both configurations.

Since we have proven (a), observe that if $j \neq 1$ and $R_{j-1} = R_j$, then $val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = y_{j-1} = val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1})$. Furthermore, recall that within the proof of (a), we showed that if either $j = 1$ or $R_{j-1} \neq R_j$, then $val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = val_{R_j}(\sigma_A^H[S_H])$. By symmetric arguments, $val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1}) = val_{R_j}(\sigma_A^H[S_H])$. Thus in both cases, $val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^{j-1}) = val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I'',D}^{j-1})$. So by the definitions of $C_{I'}$ and $C_{I''}$, R_j has the same state in $C_{I'}$ as in $C_{I''}$. Thus we have shown that:

- Every process in X_j has the same state in $C_{I'}$ as in $C_{I''}$.
- Every process in X_j is poised to access R_j in both configurations.
- R_j has the same state in $C_{I'}$ as in $C_{I''}$.

By definition, $\beta^D[j]$ contains at most one step of each process in X_j . Consequently, during both $E(C_{I'}, \beta^D[j])$ and $E(C_{I''}, \beta^D[j])$, the same subset of processes in X_j begin in the same states and perform the same operations with the same responses in the same order on the same shared object R_j which also begins in the same state. Thus it is clear that for every process $p \in X_j$, $state_p(C_{I'}, \beta^D[j]) = state_p(C_{I''}, \beta^D[j])$.

Next, since both $\sigma_{I',D}$ and $\sigma_{I'',D}$ contain the contiguous subsequence $\beta^D[j]$, observe that by definition, $\sigma_{I',D}^{j-1} \circ \beta^D[j] = \sigma_{I',D}^j$ and $\sigma_{I'',D}^{j-1} \circ \beta^D[j] = \sigma_{I'',D}^j$. Thus by the definitions of $C_{I'}$ and $C_{I''}$, for every process $p \in X_j$, $state_p(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = state_p(\sigma_A^H[S_H] \circ \sigma_{I'',D}^j)$.

By [Lemma 22](#), observe that by definition, $\sigma_{I',D}^j$ is a prefix of $\sigma_{I',D}$ that contains all the steps of processes in X_j that are in $\sigma_{I',D}$. So for every process $p \in X_j$, $state_p(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = state_p(\sigma_A^H[S_H] \circ \sigma_{I',D})$. By symmetric arguments, for every process $p \in X_j$, $state_p(\sigma_A^H[S_H] \circ \sigma_{I'',D}^j) = state_p(\sigma_A^H[S_H] \circ \sigma_{I'',D})$. Thus for every process $p \in X_j$, $state_p(\sigma_A^H[S_H] \circ \sigma_{I',D}) = state_p(\sigma_A^H[S_H] \circ \sigma_{I'',D})$.

Finally, by definition, process z_j^D is in X_j . Therefore $state_{z_j^D}(\sigma_A^H[S_H] \circ \sigma_{I',D}) = state_{z_j^D}(\sigma_A^H[S_H] \circ \sigma_{I'',D})$.

(c): Let C'_S and C'_{S_H} be the configurations at the end of $E(\sigma_A^H[S])$ and $E(\sigma_A^H[S_H])$ respectively.

By [Lemma 21](#), $\sigma_A^H[0..2^n-1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n-1]) = S_H$. So by [Invariant \(I3\)](#), for every process $p \in S \subseteq S_H$, $state_p(\sigma_A^H[S]) = state_p(\sigma_A^H[S_H])$. Furthermore, since $F(\sigma_A^H[S_H]) \subseteq S \subseteq S_H$, $\sigma_A^H[S] \neq \perp$ by [Invariant \(I2\)](#). So [Lemma 21](#) also asserts that for every integer $j'' \in \{1, 2, \dots, m\}$, $val_{R_{j''}}(\sigma_A^H[S]) = val_{R_{j''}}(\sigma_A^H[S_H])$. Thus by the definitions of C'_S and C'_{S_H} :

- Every process in S has the same state in C'_S as in C'_{S_H} .
- For every integer $j'' \in \{1, 2, \dots, m\}$, $R_{j''}$ has the same state in C'_S as in C'_{S_H} .

By the definition of $\sigma_{I',D}$, since both $z_{j'}^D \in S$ for every $j' \in I'$ and $A_{j''} \cup B_{j''}^D \subseteq S_\alpha \subseteq S$ for every integer $j'' \in \{1, 2, \dots, m\}$, S contains every process that has any steps in $\sigma_{I',D}$. Furthermore, by definition, $\sigma_{I',D}$ contains at most one step of each process in X . Thus by the definitions of C'_S and C'_{S_H} and by [Lemma 21](#), every shared object that is accessed during either $E(C'_S, \sigma_{I',D})$ or $E(C'_{S_H}, \sigma_{I',D})$ is in $\{R_1, R_2, \dots, R_m\}$.

Consequently, observe that during both $E(C'_S, \sigma_{I',D}^j)$ and $E(C'_{S_H}, \sigma_{I',D}^j)$, the same subset of processes in $S \subseteq S_H$ begin in the same states and perform the same operations with the same responses in the same order on the same shared objects which also begin in the same states. Thus it is clear that for every process $p \in S$, $state_p(C'_S, \sigma_{I',D}^j) = state_p(C'_{S_H}, \sigma_{I',D}^j)$. Furthermore, $val_{R_j}(C'_S, \sigma_{I',D}^j) = val_{R_j}(C'_{S_H}, \sigma_{I',D}^j)$. So by the definitions of C'_S and C'_{S_H} :

- For every process $p \in S$, $state_p(\sigma_A^H[S] \circ \sigma_{I',D}^j) = state_p(\sigma_A^H[S_H] \circ \sigma_{I',D}^j)$.
- $val_{R_j}(\sigma_A^H[S] \circ \sigma_{I',D}^j) = val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j)$.

Finally, since we have proven (a), $val_{R_j}(\sigma_A^H[S_H] \circ \sigma_{I',D}^j) = y_j$. Thus $val_{R_j}(\sigma_A^H[S] \circ \sigma_{I',D}^j) = y_j$.
(d): Immediately follows from (b) and (c). \square

Next, let $S_F = S_\alpha \cup F(\sigma_A^H[S_H])$. Since $S_\alpha \subseteq X$ and $S_H = F(\sigma_A^H[S_H]) \cup X$, $F(\sigma_A^H[S_H]) \subseteq S_F \subseteq S_H$. By [Lemma 21](#), $\sigma_A^H[0..2^n - 1]$ is $(i-1)$ -compliant, so:

- By Invariant (I2), $\sigma_A^H[S_F] \neq \perp$.
- By Invariant (I1), $P(\sigma_A^H[S_F]) \subseteq S_F$.
- By Invariant (I4) $F(\sigma_A^H[S_F]) = F(\sigma_A^H[S_H])$.

Thus observe that to avoid violating deadlock freedom, there must exist a schedule σ_F such that:

- σ_F begins with exactly one crash step of every process in S_α , and contains no other crash steps.
- σ_F contains only steps of processes in $S_\alpha = S_F \setminus F(\sigma_A^H[S_F])$, i.e., $P(\sigma_F) = S_\alpha$.
- During $E(\sigma_A^H[S_F] \circ \sigma_\alpha \circ \sigma_F)$, every process in S_F begins and then completes a super-passage, i.e., $F(\sigma_A^H[S_F] \circ \sigma_\alpha \circ \sigma_F) = S_F$.

Let C_F be the configuration at the end of $E(\sigma_A^H[S_F] \circ \sigma_\alpha)$. Then let \mathcal{R}_F be the set of every shared object that is accessed during $E(C_F, \sigma_F)$ (after the crash steps of every process in S_α at the beginning of σ_F). Next, let \mathcal{D} be the set of every process $p \in X \setminus S_\alpha$ such that there exists a shared object $R \in \mathcal{R}_F$ such that either p owns R , or $last_R(\sigma_A^H[S_H]) = p$.

LEMMA 24. $|\mathcal{D}| \leq 2w|S_\alpha| = \delta|\cup_j V_j|$.

PROOF. First, consider each shared object $R \in \mathcal{R}_F$ such that R is owned by a process in S_α . Since $\mathcal{D} \subseteq X \setminus S_\alpha$, the owner of R is not in \mathcal{D} . Furthermore, by [Lemma 21](#), $\sigma_A^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_A^H[0..2^n - 1]) = S_H$, so by Invariant (I8), since the owner of R is in $S_\alpha \subseteq X \subseteq S_H \setminus F(\sigma_A^H[S_H])$, R cannot be accessed by any process in $X \setminus S_\alpha$ during $E(\sigma_A^H[S_H])$. So $last_R(\sigma_A^H[S_H]) \notin \mathcal{D}$. Thus intuitively, each shared object $R \in \mathcal{R}_F$ that is owned by a process in S_α does not contribute any processes to \mathcal{D} .

So it suffices to consider the shared objects in \mathcal{R}_F that are not owned by any process in S_α . By (A1), each process accesses at most w shared objects that it does not own during a passage (because a process must incur an RMR whenever it accesses a shared object that it does not own for the first time). Thus there are at most $w|S_\alpha|$ shared objects in \mathcal{R}_F that are not owned by any process in S_α . Consequently, $|\mathcal{D}| \leq 2w|S_\alpha| = \delta|\cup_j V_j|$. \square

Let $\mathcal{Z} = \bigcup_{j \in I^D} z_j^D$, $S_\beta = S_\alpha \cup \mathcal{Z}$, and $S_B = S_\beta \cup F(\sigma_A^H[S_H]) = \mathcal{Z} \cup S_\alpha \cup F(\sigma_A^H[S_H])$. Note that $\mathcal{Z} \cap \mathcal{D} = \emptyset$, and $\mathcal{Z} \subseteq S_\beta \subseteq X$ since both $S_\alpha \subseteq X$ and $\mathcal{Z} \subseteq X$. Then, since $S_\beta \subseteq X$, $S_B = S_\beta \cup F(\sigma_A^H[S_H])$, and $S_H = F(\sigma_A^H[S_H]) \cup X$, it is clear that $F(\sigma_A^H[S_H]) \subseteq S_B \subseteq S_H$.

We now construct a new array $\sigma_B^H[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_B$, then $\sigma_B^H[S] = \perp$; otherwise $\sigma_B^H[S] = \sigma_A^H[S]$.

LEMMA 25. *This new array $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$. Furthermore, for every integer $j \in \{1, 2, \dots, m\}$:*

- For every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_B^H[S])$.
- The owner of R_j is not in $S_B \setminus F(\sigma_B^H[S_B])$.
- For every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, $\text{val}_{R_j}(\sigma_B^H[S]) = \text{val}_{R_j}(\sigma_B^H[S_B])$.

In addition, for every shared object $R \in \mathcal{R}_F$,

- The owner of R is not in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$.
- For every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$ and $S_\alpha \subseteq S$, $\text{val}_R(\sigma_B^H[S]) = \text{val}_R(\sigma_B^H[S_B])$.

PROOF. By Lemma 21, $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_A^H[0..2^n - 1]) = S_H$. By construction, $\sigma_B^H[0..2^n - 1]$ is simply a modification of $\sigma_A^H[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains any process in $S_H \setminus S_B$ has had $\sigma_B^H[S]$ set to \perp , where $F(\sigma_A^H[S_H]) \subseteq S_B \subseteq S_H$. It suffices to observe that every invariant still holds with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$, and thus $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$.

Furthermore, by Lemma 21, for every integer $j \in \{1, 2, \dots, m\}$:

- For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_A^H[S])$.
- The owner of R_j is not in $S_H \setminus F(\sigma_A^H[S_H])$.
- For every set $S \subseteq \mathcal{P}$ such that $\sigma_A^H[S] \neq \perp$, $\text{val}_{R_j}(\sigma_A^H[S]) = \text{val}_{R_j}(\sigma_A^H[S_H])$.

By the construction of $\sigma_B^H[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, $\sigma_B^H[S] = \sigma_A^H[S]$. Thus for every integer $j \in \{1, 2, \dots, m\}$ and every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_B^H[S])$.

By Lemma 21, $\sigma_A^H[0..2^n - 1]$ is $(i - 1)$ -compliant. So by Invariant (I4), $F(\sigma_A^H[S_H]) = F(\sigma_A^H[S_B]) = F(\sigma_B^H[S_B])$. Then, since $S_B \subseteq S_H$, for every integer $j \in \{1, 2, \dots, m\}$, the owner of R_j is not in $S_B \setminus F(\sigma_B^H[S_B])$.

Next, for every integer $j \in \{1, 2, \dots, m\}$ and every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$,

$$\begin{aligned} \text{val}_{R_j}(\sigma_B^H[S]) &= \text{val}_{R_j}(\sigma_A^H[S]) \\ &= \text{val}_{R_j}(\sigma_A^H[S_H]) \\ &= \text{val}_{R_j}(\sigma_A^H[S_B]) \\ &= \text{val}_{R_j}(\sigma_B^H[S_B]) \end{aligned}$$

Thus we have proven that for every integer $j \in \{1, 2, \dots, m\}$:

- For every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_B^H[S])$.
- The owner of R_j is not in $S_B \setminus F(\sigma_B^H[S_B])$.
- For every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, $\text{val}_{R_j}(\sigma_B^H[S]) = \text{val}_{R_j}(\sigma_B^H[S_B])$.

Next, by definition, $\mathcal{D} \subseteq \mathcal{P}$ is the set of every process $p \in X \setminus S_\alpha$ such that there exists a shared object $R \in \mathcal{R}_F$ such that either p owns R , or $\text{last}_R(\sigma_A^H[S_H]) = p$. By construction, $\mathcal{Z} \subseteq X$ and $\mathcal{Z} \cap \mathcal{D} = \emptyset$. Thus for every process $p \in \mathcal{Z}$, there is no shared object $R \in \mathcal{R}_F$ such that either p owns R , or $\text{last}_R(\sigma_A^H[S_H]) = p$. By construction, $S_\beta = S_\alpha \cup \mathcal{Z}$, and $S_B = S_\beta \cup F(\sigma_A^H[S_H])$. Furthermore, recall that $F(\sigma_A^H[S_H]) = F(\sigma_B^H[S_B])$. So $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B])) \subseteq \mathcal{Z}$. Thus for every process $p \in S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$, there is no shared object $R \in \mathcal{R}_F$ such that either p owns R , or $\text{last}_R(\sigma_A^H[S_H]) = p$. So for every shared object $R \in \mathcal{R}_F$, the owner of R is not in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$.

It now remains to show that for every shared object $R \in \mathcal{R}_F$, and every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$ and $S_\alpha \subseteq S$, $\text{val}_R(\sigma_B^H[S]) = \text{val}_R(\sigma_B^H[S_B])$. Let R' be a shared object in \mathcal{R}_F and $S' \subseteq \mathcal{P}$ be a set such that $\sigma_B^H[S'] \neq \perp$ and $S_\alpha \subseteq S'$. It suffices to show that $\text{val}_{R'}(\sigma_B^H[S']) = \text{val}_{R'}(\sigma_B^H[S_B])$.

By [Lemma 21](#), $\sigma_A^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_A^H[0..2^n - 1]) = S_H$. So by [Invariant \(15\)](#), there is a value $y_{R'}$ such that for every set $S \subseteq \mathcal{P}$, if $\sigma_A^H[S] \neq \perp$, then:

$$\text{val}_{R'}(\sigma_A^H[S]) = \begin{cases} \text{val}_{R'}(\sigma_A^H[S_H]) & \text{if } \text{last}_{R'}(\sigma_A^H[S_H]) \in S \\ y_{R'} & \text{otherwise} \end{cases}$$

Thus it suffices to show that $\text{last}_{R'}(\sigma_A^H[S_H]) \notin S_B \setminus S'$.

Recall that for every process $p \in S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$, there is no shared object $R \in \mathcal{R}_F$ such that either p owns R , or $\text{last}_R(\sigma_A^H[S_H]) = p$. So $\text{last}_{R'}(\sigma_A^H[S_H]) \notin S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$.

Since we have already proven that $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$, by [Invariant \(12\)](#), since $\sigma_B^H[S'] \neq \perp$ and $S_\alpha \subseteq S'$, we derive that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S'$. So $S_B \setminus S' \subseteq S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$. Thus $\text{last}_{R'}(\sigma_A^H[S_H]) \notin S_B \setminus S'$.

Consequently, for every shared object $R \in \mathcal{R}_F$, and every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$ and $S_\alpha \subseteq S$, $\text{val}_R(\sigma_B^H[S]) = \text{val}_R(\sigma_B^H[S_B])$. \square

LEMMA 26. *For every set $I' \subseteq I^{\mathcal{D}}$, every integer $j \in \{1, 2, \dots, m\}$, and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$ and $z_{j'}^{\mathcal{D}} \in S$ for every $j' \in I'$:*

- Both $\text{val}_{R_j}(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}^j) = y_j$ and for every process $p \in S$, $\text{state}_p(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}^j) = \text{state}_p(\sigma_B^H[S_B] \circ \sigma_{I', \mathcal{D}}^j)$.
- For every set $I'' \subseteq I^{\mathcal{D}}$ such that both $\sigma_{I', \mathcal{D}}$ and $\sigma_{I'', \mathcal{D}}$ contain the contiguous subsequence $\beta^{\mathcal{D}}[j]$ and $z_{j'}^{\mathcal{D}} \in S$ for every $j' \in I''$, $\text{state}_{z_j^{\mathcal{D}}}(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}) = \text{state}_{z_j^{\mathcal{D}}}(\sigma_B^H[S] \circ \sigma_{I'', \mathcal{D}})$.
- If $j = m$ or $R_j \neq R_{j+1}$, then $\text{val}_{R_j}(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}) = y_j$.

PROOF. (a): Recall that by definition, $\mathcal{D} \subseteq X$. Furthermore, by [Lemma 24](#), $|\mathcal{D}| \leq \delta |\cup_j V_j|$. So by [Lemma 23\(c\)](#), for every set $I' \subseteq I^{\mathcal{D}}$, every integer $j \in \{1, 2, \dots, m\}$, and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_A^H[S_H]) \cup S_\alpha \subseteq S \subseteq S_H$ and $z_{j'}^{\mathcal{D}} \in S$ for every $j' \in I'$, both $\text{val}_{R_j}(\sigma_A^H[S] \circ \sigma_{I', \mathcal{D}}^j) = y_j$ and for every process $p \in S$, $\text{state}_p(\sigma_A^H[S] \circ \sigma_{I', \mathcal{D}}^j) = \text{state}_p(\sigma_A^H[S_H] \circ \sigma_{I', \mathcal{D}}^j)$.

Next, recall that $S_B \subseteq S_H$. Furthermore, by construction, for every set $S \subseteq S_B \subseteq S_H$, $\sigma_B^H[S] = \sigma_A^H[S]$. So by [Lemma 21](#) and [Invariant \(14\)](#), $F(\sigma_B^H[S_B]) = F(\sigma_A^H[S_H])$. Thus for every set $I' \subseteq I^{\mathcal{D}}$, every integer $j \in \{1, 2, \dots, m\}$, and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$ and $z_{j'}^{\mathcal{D}} \in S$ for every $j' \in I'$, both $\text{val}_{R_j}(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}^j) = y_j$ and for every process $p \in S$, $\text{state}_p(\sigma_B^H[S] \circ \sigma_{I', \mathcal{D}}^j) = \text{state}_p(\sigma_B^H[S_B] \circ \sigma_{I', \mathcal{D}}^j)$.

(b): Recall that $\mathcal{D} \subseteq X$ and $|\mathcal{D}| \leq \delta |\cup_j V_j|$. By [Lemma 23\(d\)](#), for every set $I' \subseteq I^{\mathcal{D}}$, every integer $j \in \{1, 2, \dots, m\}$, every set $I'' \subseteq I^{\mathcal{D}}$ such that both $\sigma_{I', \mathcal{D}}$ and $\sigma_{I'', \mathcal{D}}$ contain the contiguous subsequence $\beta^{\mathcal{D}}[j]$ and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_A^H[S_H]) \cup S_\alpha \subseteq S \subseteq S_H$ and $z_{j'}^{\mathcal{D}} \in S$

for every $j' \in I' \cup I''$, $state_{z_{j'}^D}(\sigma_A^H[S] \circ \sigma_{I',D}) = state_{z_{j'}^D}(\sigma_A^H[S] \circ \sigma_{I'',D})$. Recall that $S_B \subseteq S_H$, $F(\sigma_B^H[S_B]) = F(\sigma_A^H[S_H])$, and for every set $S \subseteq S_B \subseteq S_H$, $\sigma_B^H[S] = \sigma_A^H[S]$. Thus for every set $I' \subseteq I^D$, every integer $j \in \{1, 2, \dots, m\}$, every set $I'' \subseteq I^D$ such that both $\sigma_{I',D}$ and $\sigma_{I'',D}$ contain the contiguous subsequence $\beta^D[j]$ and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$ and $z_{j'}^D \in S$ for every $j' \in I' \cup I''$, $state_{z_j^D}(\sigma_B^H[S] \circ \sigma_{I',D}) = state_{z_j^D}(\sigma_B^H[S] \circ \sigma_{I'',D})$.

In other words, for every set $I' \subseteq I^D$, every integer $j \in \{1, 2, \dots, m\}$, and every set $S \subseteq \mathcal{P}$ such that both $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$ and $z_{j'}^D \in S$ for every $j' \in I'$, and every set $I'' \subseteq I^D$ such that both $\sigma_{I',D}$ and $\sigma_{I'',D}$ contain the contiguous subsequence $\beta^D[j]$ and $z_{j'}^D \in S$ for every $j' \in I''$, $state_{z_j^D}(\sigma_B^H[S] \circ \sigma_{I',D}) = state_{z_j^D}(\sigma_B^H[S] \circ \sigma_{I'',D})$.

(c): Since we have already proven (a), $val_{R_j}(\sigma_B^H[S] \circ \sigma_{I',D}^j) = y_j$. By definition, if $j = m$, then $\sigma_{I',D}^j = \sigma_{I',D}$, so $val_{R_j}(\sigma_B^H[S] \circ \sigma_{I',D}) = y_j$.

Thus it suffices to consider the case where $R_j \neq R_{j+1}$. Then by the ordering of the shared objects R_1, R_2, \dots, R_m , since $R_j \neq R_{j+1}$, there is no integer $j' > j$ in $\{1, 2, \dots, m\}$ such that $R_{j'} = R_j$. So by Lemma 25, for every integer $j' > j$, every process in $X_{j'} \cap S$ is not poised to access R_j at the end of $E(\sigma_B^H[S])$.

By definition, since $z_{j'}^D \in S$ for every $j' \in I'$, $\sigma_{I',D}$ contains at most one step by each process in $X \cap S$. Thus observe that every access of R_j during $E(\sigma_B^H[S] \circ \sigma_{I',D})$ also occurs during $E(\sigma_B^H[S] \circ \sigma_{I',D}^j)$. Consequently, $val_{R_j}(\sigma_B^H[S] \circ \sigma_{I',D}) = val_{R_j}(\sigma_B^H[S] \circ \sigma_{I',D}^j) = y_j$. \square

Lemma 25 asserts that $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_B^H[0..2^n - 1]) = S_B$. Furthermore, $S_\alpha \subseteq S_\beta \subseteq S_B$, so $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S_B$. Thus by Invariant (I2), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, $\sigma_B^H[S] \neq \perp$.

So for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, let C'_S be the configuration at the end of $E(\sigma_B^H[S])$ and let $I'_S \subseteq I^D$ be the largest set such that $z_{j'}^D \in S$ for every $j' \in I'_S$. Then let $\sigma'_S = \sigma_{I'_S,D}$.

Next, recall that $S_F = S_\alpha \cup F(\sigma_A^H[S_H])$ and that by Lemma 21 and Invariant (I4), $F(\sigma_A^H[S_H]) = F(\sigma_A^H[S_B]) = F(\sigma_B^H[S_B])$. Thus $S_F = S_\alpha \cup F(\sigma_B^H[S_B])$, and so $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S_F \subseteq S_B$. Furthermore, recall that by definition, since $S_F = S_\alpha \cup F(\sigma_B^H[S_B])$, $\sigma_\alpha = \sigma'_{S_F}$. Thus $E(C'_{S_F}, \sigma'_{S_F}) = E(C'_{S_F}, \sigma_\alpha)$.

LEMMA 27. *For every shared object $R \in \mathcal{R}_F$, and every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, $val_R(C'_S, \sigma'_S) = val_R(C'_{S_F}, \sigma_\alpha)$.*

PROOF. Let $S \subseteq \mathcal{P}$ be a set of processes such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$. By Lemma 25 and Invariant (I2), since $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S_F \subseteq S_B$ and $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, both $\sigma_B^H[S_F] \neq \perp$ and $\sigma_B^H[S] \neq \perp$. Thus by Lemma 25, since $S_\alpha \subseteq S_F$ and $S_\alpha \subseteq S$, for every shared object $R \in \mathcal{R}_F$, $val_R(\sigma_B^H[S_F]) = val_R(\sigma_B^H[S_B]) = val_R(\sigma_B^H[S])$.

Furthermore, by Lemma 25, since both $\sigma_B^H[S_F] \neq \perp$ and $\sigma_B^H[S] \neq \perp$, for every integer $j \in \{1, 2, \dots, m\}$, every process in $X_j \cap S_F$ is poised to access R_j at the end of $E(\sigma_B^H[S_F])$ and every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_B^H[S])$. Therefore, since σ_α contains at most one step of each process in $X \cap S_F$ and σ'_S contains at most one step of each process in $X \cap S$, the only shared objects that can be accessed during $E(C'_S, \sigma'_S)$ or $E(C'_{S_F}, \sigma_\alpha)$ are $\{R_1, R_2, \dots, R_m\}$. Thus for every shared object $R \in \mathcal{R}_F$ such that $R \neq R_j$ for any integer $j \in \{1, 2, \dots, m\}$, $val_R(\sigma_B^H[S_F] \circ \sigma_\alpha) = val_R(\sigma_B^H[S] \circ \sigma'_S)$.

On the other hand, observe that by [Lemma 26\(c\)](#), for every integer $j \in \{1, 2, \dots, m\}$, $\text{val}_{R_j}(C'_S, \sigma'_S) = \text{val}_{R_j}(C'_{S_F}, \sigma_\alpha)$. Consequently, for every shared object $R \in \mathcal{R}_F$, regardless of whether R is one of $\{R_1, R_2, \dots, R_m\}$, $\text{val}_R(\sigma_B^H[S_F] \circ \sigma_\alpha) = \text{val}_R(\sigma_B^H[S] \circ \sigma'_S)$. \square

LEMMA 28. For every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$:

(H1) For every process $p \in \mathcal{P}$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then the owner of R is not in $S_B \setminus F(\sigma_B^H[S_B])$.

(H2) In the CC model, for every process $p \in S \setminus F(\sigma_B^H[S_B])$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$ is exactly the same as at the end of $E(\sigma_B^H[S_B] \circ \sigma'_{S_B})$.

(H3) For each process $p \in S \setminus S_\alpha$, $\text{state}_p(C'_{S_B}, \sigma'_{S_B}) = \text{state}_p(C'_S, \sigma'_S)$.

(H4) Each process in $\mathcal{Z} \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$.

(H5) For each process $p \in S \setminus F(\sigma_B^H[S_B])$, p has not left the critical section during $E(\sigma_B^H[S] \circ \sigma'_S)$.

(H6) $F(\sigma_B^H[S]) = F(\sigma_B^H[S] \circ \sigma'_S)$.

PROOF. First, recall that for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, $\sigma_B^H[S] \neq \perp$. So let $S \subseteq \mathcal{P}$ be a set of processes such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and note that $\sigma_B^H[S] \neq \perp$.

Thus by [Lemma 25](#), for every integer $j \in \{1, 2, \dots, m\}$, every process in $X_j \cap S$ is poised to access R_j at the end of $E(\sigma_B^H[S])$. By definition, σ'_S contains at most one step of each process in $X \cap S$. Therefore, the only shared objects that can be accessed during $E(C'_S, \sigma'_S)$ are $\{R_1, R_2, \dots, R_m\}$. Furthermore, by [Lemma 25](#), for every integer $j \in \{1, 2, \dots, m\}$, the owner of R_j is not in $S_B \setminus F(\sigma_B^H[S_B])$. Thus for every process $p \in \mathcal{P}$, if p accesses a shared object R during $E(C'_S, \sigma'_S)$, then the owner of R is not in $S_B \setminus F(\sigma_B^H[S_B])$ (H1).

By [Lemma 25](#), $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$. So by Invariant (I9), for every process $p \in S \setminus F(\sigma_B^H[S_B])$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_B^H[S])$ is exactly the same as at the end of $E(\sigma_B^H[S_B])$.

Now recall that by the definition of X , no process in X is poised to perform a read operation at the end of $E(\sigma_B^S[S_{\max}^S])$. By construction, $\sigma_B^H[S_B] = \sigma_A^H[S_B] = \sigma_B^S[S_B]$. By [Lemma 10](#) and [Lemma 25](#), both $\sigma_B^S[0..2^n - 1]$ and $\sigma_B^H[0..2^n - 1]$ are $(i-1)$ -compliant. Thus by Invariant (I3), no process in X is poised to perform a read operation at the end of $E(\sigma_B^H[S_B])$ or $E(\sigma_B^H[S])$.

By [Lemma 22](#), for every integer $j \in \{1, 2, \dots, m\}$, A_j is a non-empty set of processes. So for every integer $j \in \{1, 2, \dots, m\}$, both $\alpha[j]$ and $\beta^D[j]$ contain at least one step by definition. Thus by definition, for every integer $j \in \{1, 2, \dots, m\}$, both σ'_S and σ'_{S_B} contain a step by a process in X_j . Therefore, since every non-read operation invalidates all cache copies on a shared object, all prior cache copies of shared objects in $\{R_1, R_2, \dots, R_m\}$ are invalidated during both $E(C'_S, \sigma'_S)$ and $E(C'_{S_B}, \sigma'_{S_B})$.

On the other hand, by definition, σ'_S and σ'_{S_B} contain at most one step by each process in $X \cap S$ and $X \cap S_B$ respectively and no other steps. So no read operations are performed during $E(C'_S, \sigma'_S)$ or $E(C'_{S_B}, \sigma'_{S_B})$. Furthermore, by [Lemma 25](#), every process in $X \cap S_B$ is poised to access one of $\{R_1, R_2, \dots, R_m\}$ at the end of $E(\sigma_B^H[S_B])$ and every process in $X \cap S$ is poised to access one of $\{R_1, R_2, \dots, R_m\}$ at the end of $E(\sigma_B^H[S])$. Thus the only shared objects that can be accessed during $E(C'_S, \sigma'_S)$ or $E(C'_{S_B}, \sigma'_{S_B})$ are $\{R_1, R_2, \dots, R_m\}$.

Consequently, during both $E(C'_S, \sigma'_S)$ and $E(C'_{S_B}, \sigma'_{S_B})$, all cache copies of shared objects in $\{R_1, R_2, \dots, R_m\}$ are invalidated, no other cache copies are invalidated, and no new cache copies are created. Thus for every process $p \in S \setminus F(\sigma_B^H[S_B])$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$ is exactly the same as at the end of $E(\sigma_B^H[S_B] \circ \sigma'_{S_B})$ (H2).

Now recall that $S_B = \mathcal{Z} \cup S_\alpha \cup F(\sigma_B^H[S_B])$, and $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$. Thus every process in $S \setminus S_\alpha$ is in $\mathcal{Z} \cup F(\sigma_B^H[S_B])$. By [Lemma 25](#), $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant. So by [Invariant \(I4\)](#), $F(\sigma_B^H[S]) = F(\sigma_B^H[S_B])$. Thus it is clear that for every process $p \in F(\sigma_B^H[S_B])$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$.

Next, recall that by definition, $\sigma'_S = \sigma_{I'_S, \mathcal{D}}$, where $I'_S \subseteq I^{\mathcal{D}}$ is the largest set such that $z_{j'}^{\mathcal{D}} \in S$ for every $j' \in I'_S$. So by [Lemma 26\(a\)](#), the state of each process in S is the same at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$ as at the end of $E(\sigma_B^H[S_B] \circ \sigma'_S)$. Note that this includes process $z_{j'}^{\mathcal{D}}$ for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$.

Furthermore, recall that by definition, $\sigma'_{S_B} = \sigma_{I'_{S_B}, \mathcal{D}}$, where $I'_{S_B} \subseteq I^{\mathcal{D}}$ is the largest set such that $z_{j'}^{\mathcal{D}} \in S_B$ for every $j' \in I'_{S_B}$. Since $\mathcal{Z} \subseteq S_B$, observe that $I'_{S_B} = I^{\mathcal{D}}$. So $I'_S \subseteq I'_{S_B}$. Thus for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$, $j' \in I'_S$ and $j' \in I'_{S_B}$. So by definition, for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$, both σ'_S and σ'_{S_B} contain the contiguous subsequence $\beta^{\mathcal{D}}[j']$. Thus by [Lemma 26\(b\)](#), for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$, $state_{z_{j'}^{\mathcal{D}}}(\sigma_B^H[S_B] \circ \sigma'_S) = state_{z_{j'}^{\mathcal{D}}}(\sigma_B^H[S_B] \circ \sigma'_{S_B})$. Consequently, since the state of each process in S is the same at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$ as at the end of $E(\sigma_B^H[S_B] \circ \sigma'_S)$, for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$, $state_{z_{j'}^{\mathcal{D}}}(\sigma_B^H[S] \circ \sigma'_S) = state_{z_{j'}^{\mathcal{D}}}(\sigma_B^H[S_B] \circ \sigma'_{S_B})$.

So by the definition of C'_S and C'_{S_B} , for every integer $j' \in I^{\mathcal{D}}$ such that $z_{j'}^{\mathcal{D}} \in S$, $state_{z_{j'}^{\mathcal{D}}}(C'_S, \sigma'_S) = state_{z_{j'}^{\mathcal{D}}}(C'_{S_B}, \sigma'_{S_B})$. Thus by the definition of \mathcal{Z} , observe that for each process $p \in S \cap \mathcal{Z}$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$.

Therefore we have shown that:

- Every process in $S \setminus S_\alpha$ is in $\mathcal{Z} \cup F(\sigma_B^H[S_B])$.
- For every process $p \in F(\sigma_B^H[S_B])$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$.
- For each process $p \in S \cap \mathcal{Z}$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$.

It immediately follows that for each process $p \in S \setminus S_\alpha$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$ ([H3](#)).

Since $\sigma_B^H[S] \neq \perp$, by construction, $\sigma_{old}[S] \neq \perp$. By [Lemma 8 \(S3\)](#), for each process $p \in S \setminus F(\sigma_{old}[S_{max}^{old}])$, p incurs an RMR at the end of $E(\sigma_A^S[S] \circ p)$. By construction and [Lemma 9](#) ([Invariant \(I4\)](#)), $\sigma_B^H[S] = \sigma_A^H[S] = \sigma_B^S[S] = \sigma_A^S[S]$, and $F(\sigma_B^H[S_B]) = F(\sigma_A^S[S_B]) = F(\sigma_A^S[S_{max}^{old}]) = F(\sigma_{old}[S_{max}^{old}])$. Thus for each process $p \in S \setminus F(\sigma_B^H[S_B])$, p incurs an RMR at the end of $E(\sigma_B^H[S] \circ p)$.

By definition, $S_B = S_\beta \cup F(\sigma_B^H[S_B])$. So since $S \subseteq S_B$, $S \setminus F(\sigma_B^H[S_B]) = S_\beta \cap S$. By construction, apart from the steps of processes in S_α , σ'_S contains exactly one non-crash step of each process in $\mathcal{Z} \cap S$. Therefore:

- In the DSM model, every process in $\mathcal{Z} \cap S$ is poised to access a shared object it does not own in C'_S , and so every process in $\mathcal{Z} \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$.
- In the CC model, every process in $\mathcal{Z} \cap S$ is poised to perform a non-read operation or read a shared object that it does not have a valid cache copy of in C'_S , and so every process in $\mathcal{Z} \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$.

Thus in both the DSM and CC models, each process in $\mathcal{Z} \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$ ([H4](#)).

By [Lemma 25](#), $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant, so by [Invariant \(I7\)](#), each process that is not in $F(\sigma_B^H[S])$ does not enter the critical section during $E(\sigma_B^H[S])$. Recall that σ'_S consists of exactly one non-crash step of each process in $S_\beta \cap S = S \setminus F(\sigma_B^H[S_B])$ and no other steps. Thus, with only one step, although a process could enter the critical section during $E(C'_S, \sigma'_S)$, it cannot have

taken any steps within the critical section. Thus by (A2), no process can leave the critical section during $E(C'_S, \sigma'_S)$. So for each process $p \in S \setminus F(\sigma_B^H[S_B])$, p has not left the critical section during $E(\sigma_B^H[S] \circ \sigma'_S)$ (H5). Therefore, since no process in $S \setminus F(\sigma_B^H[S_B])$ has left the critical section during $E(\sigma_B^H[S] \circ \sigma'_S)$, no process in $S \setminus F(\sigma_B^H[S_B])$ has completed its super-passage during $E(\sigma_B^H[S] \circ \sigma'_S)$. Thus $F(\sigma_B^H[S]) = F(\sigma_B^H[S] \circ \sigma'_S)$ (H6). \square

Next, recall that there exists a schedule σ_F such that:

- σ_F begins with exactly one crash step of every process in S_α , and contains no other crash steps.
- σ_F contains only steps of processes in $S_\alpha = S_F \setminus F(\sigma_A^H[S_F])$.
- During $E(\sigma_A^H[S_F] \circ \sigma_\alpha \circ \sigma_F)$, every process in S_F begins and then completes a super-passage, i.e., $F(\sigma_A^H[S_F] \circ \sigma_\alpha \circ \sigma_F) = S_F$.

By construction, $\sigma_B^H[S_F] = \sigma_A^H[S_F]$ and $F(\sigma_A^H[S_B]) = F(\sigma_B^H[S_B])$. By Lemma 21 and Invariant (I4), $F(\sigma_A^H[S_F]) = F(\sigma_A^H[S_B]) = F(\sigma_B^H[S_B])$. Thus:

- σ_F begins with exactly one crash step of every process in S_α , and contains no other crash steps.
- σ_F contains only steps of processes in $S_\alpha = S_F \setminus F(\sigma_B^H[S_B])$.
- During $E(\sigma_B^H[S_F] \circ \sigma_\alpha \circ \sigma_F)$, every process in S_F begins and then completes a super-passage, i.e., $F(\sigma_B^H[S_F] \circ \sigma_\alpha \circ \sigma_F) = S_F$.

Further recall that by definition, C_F is the configuration at the end of $E(\sigma_A^H[S_F] \circ \sigma_\alpha)$, and \mathcal{R}_F is the set of every shared object that is accessed during $E(C_F, \sigma_F)$ (after the crash steps of every process in S_α at the beginning of σ_F).

Now for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, let C''_S be the configuration at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$. Then note that since $E(\sigma_A^H[S_F] \circ \sigma_\alpha) = E(\sigma_B^H[S_F] \circ \sigma'_S)$, $C_F = C''_{S_F}$.

LEMMA 29. *For every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, during both $E(C''_S, \sigma_F)$ and $E(C''_{S_F}, \sigma_F) = E(C_F, \sigma_F)$, the same set of processes (namely S_α) crash, then perform the same operations with the same responses in the same order on the same set of shared objects (namely \mathcal{R}_F) which begin in the same states, and so must reach the same resulting states.*

PROOF. By Lemma 27, for every shared object $R \in \mathcal{R}_F$ and every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, R has the same state in $C_F = C''_{S_F}$ as in C''_S . Furthermore, by the definition of σ_F , σ_F begins with a crash step of every process in $P(\sigma_F) = S_\alpha$. The lemma immediately follows. \square

We now construct a new array $\sigma_C^H[0..2^n - 1]$ such that for every set $S \subseteq \mathcal{P}$, if $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, then $\sigma_C^H[S] = \sigma_B^H[S] \circ \sigma'_S \circ \sigma_F$; otherwise $\sigma_C^H[S] = \perp$.

LEMMA 30. *This new array $\sigma_C^H[0..2^n - 1]$ is i -compliant with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$ and $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$.*

PROOF. For every set $S \subseteq \mathcal{P}$, if $\sigma_C^H[S] \neq \perp$, then $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and by construction, $\sigma_C^H[S] = \sigma_B^H[S] \circ \sigma'_S \circ \sigma_F$. By Lemma 25, $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I1), $P(\sigma_B^H[S]) \subseteq S$. By the definition of σ'_S , σ'_S contains only steps of processes in S . By the definition of σ_F , σ_F contains only steps of processes in $S_\alpha \subseteq S$. Thus $P(\sigma_C^H[S]) \subseteq S$ (Invariant (I1)).

Now for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, consider $F(\sigma_C^H[S]) = F(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$. By Lemma 28 (H6), $F(\sigma_B^H[S]) = F(\sigma_B^H[S] \circ \sigma'_S)$. By definition, every process in S_α completes its super-passage during $E(C_F, \sigma_F)$. So by Lemma 29, every process in S_α also completes its super-passage during $E(C''_S, \sigma_F)$. Thus $F(\sigma_C^H[S]) = F(\sigma_B^H[S] \circ \sigma'_S) \cup S_\alpha = F(\sigma_B^H[S]) \cup S_\alpha$. Therefore, $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$.

Furthermore, by [Lemma 25](#) and Invariants [\(I2\)](#) and [\(I4\)](#), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, $F(\sigma_B^H[S]) = F(\sigma_B^H[S_B])$, and so $F(\sigma_C^H[S]) = F(\sigma_B^H[S]) \cup S_\alpha = F(\sigma_B^H[S_B]) \cup S_\alpha = F(\sigma_C^H[S_B])$.

By construction, for every set $S \subseteq \mathcal{P}$, if $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, then $\sigma_C^H[S] = \sigma_B^H[S] \circ \sigma'_S \circ \sigma_F$; otherwise $\sigma_C^H[S] = \perp$. By [Lemma 25](#), $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$. So by Invariant [\(I2\)](#), for every set $S \subseteq \mathcal{P}$, $\sigma_B^H[S] \neq \perp$ if and only if $F(\sigma_B^H[S_B]) \subseteq S \subseteq S_B$. Thus for every set $S \subseteq \mathcal{P}$, if $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, then $\sigma_B^H[S] \neq \perp$ and $\sigma_C^H[S] = \sigma_B^H[S] \circ \sigma'_S \circ \sigma_F \neq \perp$. Then, since we have already proven that $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$, for every set $S \subseteq \mathcal{P}$, $\sigma_C^H[S] \neq \perp$ if and only if $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$ (Invariant [\(I2\)](#)).

Furthermore, we have already shown that for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, $F(\sigma_C^H[S]) = F(\sigma_C^H[S_B])$. Thus, since we just proved that Invariant [\(I2\)](#) holds for $\sigma_C^H[0..2^n - 1]$ with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$ and $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$, it follows that for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, $F(\sigma_C^H[S]) = F(\sigma_C^H[S_B])$ (Invariant [\(I4\)](#)).

Next, by [Lemma 28 \(H3\)](#), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha = F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, and every process $p \in S \setminus S_\alpha$, $state_p(C'_{S_B}, \sigma'_{S_B}) = state_p(C'_S, \sigma'_S)$. Since σ_F only contains steps of processes in S_α , for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha = F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, and every process $p \in S \setminus S_\alpha$, $state_p(C'_{S_B}, \sigma'_{S_B} \circ \sigma_F) = state_p(C'_S, \sigma'_S \circ \sigma_F)$. Thus for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, and every process $p \in S \setminus S_\alpha$, $state_p(\sigma_C^H[S_B]) = state_p(\sigma_C^H[S])$.

Furthermore, we have already proven that $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$, and that Invariants [\(I2\)](#) and [\(I4\)](#) hold for $\sigma_C^H[0..2^n - 1]$ with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$. Thus for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, S_α is in both $F(\sigma_C^H[S])$ and $F(\sigma_C^H[S_B])$, and so for every process $p \in S_\alpha$, $state_p(\sigma_C^H[S_B]) = state_p(\sigma_C^H[S])$. Consequently, for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, and every process $p \in S$ regardless of whether p is in S_α , $state_p(\sigma_C^H[S_B]) = state_p(\sigma_C^H[S])$ (Invariant [\(I3\)](#)).

Next, by [Lemma 29](#), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, and every shared object $R \in \mathcal{R}_F$, $val_R(\sigma_C^H[S]) = val_R(\sigma_C^H[S_F])$. Furthermore, only shared objects in \mathcal{R}_F are accessed during $E(C'_S, \sigma_F)$. So for every integer $j \in \{1, 2, \dots, m\}$, if $R_j \notin \mathcal{R}_F$, then $val_{R_j}(C'_S, \sigma'_S \circ \sigma_F) = val_{R_j}(C'_S, \sigma'_S)$ and $val_{R_j}(C'_{S_B}, \sigma'_{S_B} \circ \sigma_F) = val_{R_j}(C'_{S_B}, \sigma'_{S_B})$. Thus by [Lemma 26\(c\)](#), if $j = m$ or $R_j \neq R_{j+1}$, then $val_{R_j}(C'_S, \sigma'_S \circ \sigma_F) = val_{R_j}(C'_{S_B}, \sigma'_{S_B} \circ \sigma_F) = y_j$. Otherwise, there must exist an integer $j' > j$ such that $R_j = R_{j'}$ and either $j' = m$ or $R_{j'} \neq R_{j'+1}$. Then by [Lemma 26\(c\)](#), $val_{R_j}(C'_S, \sigma'_S \circ \sigma_F) = val_{R_j}(C'_{S_B}, \sigma'_{S_B} \circ \sigma_F) = y_{j'}$. So in both cases, $val_{R_j}(C'_S, \sigma'_S \circ \sigma_F) = val_{R_j}(C'_{S_B}, \sigma'_{S_B} \circ \sigma_F)$, i.e., $val_{R_j}(\sigma_C^H[S]) = val_{R_j}(\sigma_C^H[S_B])$.

Thus we have shown that for every shared object $R \in \mathcal{R}$ such that either $R \in \mathcal{R}_F$ or R is one of $\{R_1, R_2, \dots, R_m\}$, and every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, $val_R(\sigma_C^H[S]) = val_R(\sigma_C^H[S_B])$. Then since we have already proven that Invariant [\(I2\)](#) holds for $\sigma_C^H[0..2^n - 1]$ with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$, for every shared object $R \in \mathcal{R}$ such that either $R \in \mathcal{R}_F$ or R is one of $\{R_1, R_2, \dots, R_m\}$, and every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, $val_R(\sigma_C^H[S]) = val_R(\sigma_C^H[S_B])$. So for every shared object $R \in \mathcal{R}$ such that either $R \in \mathcal{R}_F$ or R is one of $\{R_1, R_2, \dots, R_m\}$, regardless of $last_R(\sigma_C^H[S_B])$, for $y_R = val_R(\sigma_C^H[S_B])$, we have that for every set $S \subseteq \mathcal{P}$, if $\sigma_C^H[S] \neq \perp$, then:

$$val_R(\sigma_C^H[S]) = \begin{cases} val_R(\sigma_C^H[S_B]) & \text{if } last_R(\sigma_C^H[S_B]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Now consider the shared objects that are not in \mathcal{R}_F and not one of $\{R_1, R_2, \dots, R_m\}$. Recall that these shared objects are not accessed during $E(C'_S, \sigma'_S \circ \sigma_F)$ for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$. By [Lemma 25](#), $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$. So by Invariant [\(I5\)](#), for each such shared object R , there exists a value y_R such that for every set $S \subseteq \mathcal{P}$,

if $\sigma_B^H[S] \neq \perp$, then:

$$\text{val}_R(\sigma_B^H[S]) = \begin{cases} \text{val}_R(\sigma_B^H[S_B]) & \text{if } \text{last}_R(\sigma_B^H[S_B]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Then, since each such shared object R is not accessed during $E(C'_S, \sigma'_S \circ \sigma_F)$ for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_C^H[S_B]) \subseteq S \subseteq S_B$, and we have already proven that Invariant (I2) holds for $\sigma_C^H[0..2^n - 1]$ with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$:

$$\text{val}_R(\sigma_C^H[S]) = \begin{cases} \text{val}_R(\sigma_C^H[S_B]) & \text{if } \text{last}_R(\sigma_C^H[S_B]) \in S \\ y_R & \text{otherwise} \end{cases}$$

Consequently, for every shared object $R \in \mathcal{R}$, regardless of whether $R \in \mathcal{R}_F$ and whether R is one of $\{R_1, R_2, \dots, R_m\}$, there is a value y_R such that for every set $S \subseteq \mathcal{P}$, if $\sigma_C^H[S] \neq \perp$, then:

$$\text{val}_R(\sigma_C^H[S]) = \begin{cases} \text{val}_R(\sigma_C^H[S_B]) & \text{if } \text{last}_R(\sigma_C^H[S_B]) \in S \\ y_R & \text{otherwise} \end{cases}$$

So Invariant (I5) holds for $\sigma_C^H[0..2^n - 1]$.

By Lemma 25, $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant. So by Invariant (I6), for every set $S \subseteq \mathcal{P}$ with $\sigma_B^H[S] \neq \perp$, during $E(\sigma_B^H[S])$, each process crashes at most once and each process that is not in $F(\sigma_B^H[S])$ never crashes.

By definition, for every set $S \subseteq \mathcal{P}$ with $\sigma_C^H[S] \neq \perp$, σ'_S does not contain any crash steps. Furthermore, σ_F contains exactly one crash step for each process in S_α and no other crash steps.

We have already proven that $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$ and that Invariant (I4) holds for $\sigma_C^H[0..2^n - 1]$. By Lemma 25, Invariant (I4) also holds for $\sigma_B^H[0..2^n - 1]$. So for every set $S \subseteq \mathcal{P}$ with $\sigma_C^H[S] \neq \perp$, $F(\sigma_C^H[S]) = F(\sigma_B^H[S]) \cup S_\alpha$.

So for every process $p \notin F(\sigma_C^H[S_B])$, p never crashes during $E(\sigma_C^H[S])$. Furthermore, recall that $S_\alpha \cap F(\sigma_B^H[S]) = \emptyset$, so processes in S_α never crash during $E(\sigma_B^H[S])$, and thus crash at most once during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$. Moreover, by definition, processes in $F(\sigma_B^H[S])$ do not have any steps in either σ'_S or σ_F , so they also still crash at most once during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$. Consequently, for every set $S \subseteq \mathcal{P}$ with $\sigma_C^H[S] \neq \perp$, during $E(\sigma_C^H[S])$, each process crashes at most once, and each process that is not in $F(\sigma_C^H[S])$ never crashes (Invariant (I6)).

Now suppose, for contradiction, that for some set $S \subseteq \mathcal{P}$ with $\sigma_C^H[S] \neq \perp$, some process p that is not in $F(\sigma_C^H[S])$ enters the critical section during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$. Since we have already proven that Invariant (I1) holds for $\sigma_C^H[0..2^n - 1]$, $p \in S$. Furthermore, we have also already shown that $F(\sigma_C^H[S]) = F(\sigma_B^H[S]) \cup S_\alpha$, so $p \notin F(\sigma_B^H[S]) \cup S_\alpha$.

By Lemma 25, $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant. So by Invariant (I7), since $p \notin F(\sigma_B^H[S])$, p does not enter the critical section during $E(\sigma_B^H[S])$. Furthermore, by the definition of σ_F , since $p \notin S_\alpha$, σ_F contains no steps of p . Thus p must be one of the processes that take a step during $E(C'_S, \sigma'_S)$, and must enter the critical section with this one step. By (A2), a process that enters the critical section cannot leave the critical section before incurring an RMR within the critical section. Thus, since σ_F contains no steps of $p \notin S_\alpha$, p remains in the critical section throughout $E(C'_S, \sigma_F)$.

Now consider the processes in S_α . Since $\sigma_B^H[0..2^n - 1]$ is $(i-1)$ -compliant and $S_\alpha \cap F(\sigma_B^H[S]) = \emptyset$, by Invariant (I7), the processes in S_α do not enter the critical section during $E(\sigma_B^H[S])$. Since p is in the critical section in C'_S and σ'_S contains at most one step of each process, observe that by (A2), to avoid violating mutual exclusion, each process in S_α cannot enter the critical section with its at most one step taken during $E(C'_S, \sigma'_S)$. Furthermore, since p remains in the critical section throughout

$E(C'_S, \sigma_F)$, observe that to avoid violating mutual exclusion, the processes in S_α also do not enter the critical section during $E(C'_S, \sigma_F)$. Thus, by the definitions of C'_S and C''_S , the processes in S_α do not enter the critical section during $E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$.

However, we have already shown that $F(\sigma_C^H[S]) = F(\sigma_B^H[S]) \cup S_\alpha$. Thus we have that during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$, every process in S_α completes its super-passage without entering the critical section — a contradiction. Consequently, for every set $S \subseteq \mathcal{P}$ with $\sigma_C^H[S] \neq \perp$, each process that is not in $F(\sigma_C^H[S])$ does not enter the critical section during $E(\sigma_C^H[S])$ (Invariant (I7)).

Next, by Lemma 28 (H1), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and for every shared object R that is accessed during $E(C'_S, \sigma'_S)$, the owner of R is not in $S_B \setminus F(\sigma_B^H[S_B])$. Furthermore, by the definition of \mathcal{R}_F , and Lemma 29, for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, only shared objects in \mathcal{R}_F are accessed during $E(C'_S, \sigma_F)$. By Lemma 25, for every shared object $R \in \mathcal{R}_F$, the owner of R is not in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$. Thus for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and for every shared object R that is accessed during $E(C'_S, \sigma'_S \circ \sigma_F)$, the owner of R is not in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$. Consequently, for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, during $E(C'_S, \sigma'_S \circ \sigma_F)$, each shared object $R \in \mathcal{R}$ cannot be accessed if the owner of R is in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$.

By Lemma 25, Invariant (I8) holds for $\sigma_B^H[0..2^n - 1]$ and $S_{\max}(\sigma_B^H[0..2^n - 1]) = S_B$. So in the DSM model, for every set $S \subseteq \mathcal{P}$ such that $\sigma_B^H[S] \neq \perp$, during $E(\sigma_B^H[S])$, each shared object $R \in \mathcal{R}$ can only be accessed by its owner if the owner of R is in $S_B \setminus F(\sigma_B^H[S_B])$. Thus for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$, each shared object $R \in \mathcal{R}$ can only be accessed by its owner if the owner of R is in $S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$. Now recall that $F(\sigma_C^H[S_B]) = S_\alpha \cup F(\sigma_B^H[S_B])$. Thus $S_B \setminus F(\sigma_C^H[S_B]) = S_B \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$. Consequently, in the DSM model, for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, during $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$, each shared object $R \in \mathcal{R}$ can only be accessed by its owner if the owner of R is in $S_B \setminus F(\sigma_C^H[S_B])$ (Invariant (I8)).

Next, by Lemma 28 (H2), in the CC model, for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and every process $p \in S \setminus F(\sigma_B^H[S_B])$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_B^H[S] \circ \sigma'_S)$ is exactly the same as at the end of $E(\sigma_B^H[S_B] \circ \sigma'_S)$. By Lemma 29, for every shared object $R \in \mathcal{R}$, a non-read operation is performed on R during $E(C'_S, \sigma_F)$ if and only if it is also performed on R during $E(C''_S, \sigma_F)$. By definition, σ_F contains only steps of processes in S_α . So for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, and every process $p \in S \setminus (F(\sigma_B^H[S_B]) \cup S_\alpha)$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_C^H[S]) = E(\sigma_B^H[S] \circ \sigma'_S \circ \sigma_F)$ is exactly the same as at the end of $E(\sigma_C^H[S_B]) = E(\sigma_B^H[S_B] \circ \sigma'_S \circ \sigma_F)$.

Now recall that we have already proven that $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$, and that Invariant (I2) holds for $\sigma_C^H[0..2^n - 1]$ with $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$. Thus for every set $S \subseteq \mathcal{P}$ such that $\sigma_C^H[S] \neq \perp$, and every process $p \in S \cap (S_B \setminus F(\sigma_C^H[S_B]))$, the set of shared objects that p has valid cache copies of at the end of $E(\sigma_C^H[S])$ is exactly the same as at the end of $E(\sigma_C^H[S_B])$ (Invariant (I9)).

Finally, by Lemma 25, $\sigma_B^H[0..2^n - 1]$ is $(i - 1)$ -compliant, so by Invariant (I10), for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_B^H[S])$, if $\sigma_B^H[S] \neq \perp$, then p incurs at least $i - 1$ RMRs during $E(\sigma_B^H[S])$. By Lemma 28 (H4), for every set $S \subseteq \mathcal{P}$ such that $F(\sigma_B^H[S_B]) \cup S_\alpha \subseteq S \subseteq S_B$, each process in $\mathcal{Z} \cap S$ incurs exactly one RMR during $E(C'_S, \sigma'_S)$. By construction, for every set $S \subseteq \mathcal{P}$, if $\sigma_C^H[S] \neq \perp$, then $\sigma_C^H[S] = \sigma_B^H[S] \circ \sigma'_S \circ \sigma_F$, i.e., every process in $\mathcal{Z} \cap S = S \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$ incurs at least one more RMR during $E(\sigma_C^H[S])$ than during $E(\sigma_B^H[S])$. Thus for every set $S \subseteq \mathcal{P}$

and every process $p \in S \setminus (S_\alpha \cup F(\sigma_B^H[S_B]))$, if $\sigma_C^H[S] \neq \perp$, then p incurs at least i RMRs during $E(\sigma_C^H[S])$.

Now recall that we have already proven that $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$, and that Invariant (I4) holds for $\sigma_C^H[0..2^n - 1]$. So for every set $S \subseteq \mathcal{P}$, if $\sigma_C^H[S] \neq \perp$, then $F(\sigma_C^H[S]) = F(\sigma_B^H[S_B]) \cup S_\alpha$. Thus $S \setminus (S_\alpha \cup F(\sigma_B^H[S_B])) = S \setminus F(\sigma_C^H[S])$. Therefore for every set $S \subseteq \mathcal{P}$ and every process $p \in S \setminus F(\sigma_C^H[S])$, if $\sigma_C^H[S] \neq \perp$, then p incurs at least i RMRs during $E(\sigma_C^H[S])$ (Invariant (I10)). \square

Finally, we terminate this i -th iteration by setting $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^H[0..2^n - 1]$.

B PROOF OF LEMMA 6

Recall that, for every non-negative integer i , if $\sigma_{round}[i, 0..2^n - 1]$ is i -compliant, then $S_{max}^i = S_{max}(\sigma_{round}[i, 0..2^n - 1])$, and let $n_i = |S_{max}^i \setminus F(\sigma_{round}[i, S_{max}^i])|$.

LEMMA 31. *For every non-negative integer i , if $\sigma_{round}[i, 0..2^n - 1]$ has non- \perp entries, then $\sigma_{round}[i, 0..2^n - 1]$ is i -compliant.*

PROOF. If $i = 0$, then every entry of $\sigma_{round}[0, 0..2^n - 1]$ is the empty schedule. Clearly, the array $\sigma_{round}[0, 0..2^n - 1]$ is 0-compliant.

So suppose $i > 0$. Thus if $\sigma_{round}[i, 0..2^n - 1]$ has non- \perp entries, then either $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^L[0..2^n - 1]$, $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^G[0..2^n - 1]$, or $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^H[0..2^n - 1]$. The lemma immediately follows from Lemma 15, Lemma 20, and Lemma 30. \square

We must prove that for every positive integer i , if $\sigma_{round}[i, 0..2^n - 1]$ is i -compliant, then $n_i \geq n_{i-1}/(64w^{d+1}) - 2$.

Since $\sigma_{round}[i, 0..2^n - 1]$ is i -compliant, either $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^L[0..2^n - 1]$, $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^G[0..2^n - 1]$, or $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^H[0..2^n - 1]$.

Case 1. $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^L[0..2^n - 1]$.

Then by Lemma 15, $S_{max}(\sigma_C^L[0..2^n - 1]) = S_{max}^L$. Thus $n_i = |S_{max}^L \setminus F(\sigma_C^L[S_{max}^L])|$.

Now recall that in the construction of $\sigma_C^L[0..2^n - 1]$, we checked whether there exists a process $p \in S_I \setminus F(\sigma_B^L[S_I])$ such that p is within the critical section at the end of $E(\sigma_B^L[S_I])$. If such a process p exists, then we set $\sigma_C^L[0..2^n - 1]$ to be a simple modification of $\sigma_B^L[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains p has had $\sigma_C^L[S]$ set to \perp ; otherwise we set $\sigma_C^L[0..2^n - 1] = \sigma_B^L[0..2^n - 1]$. By Lemma 14, $S_{max}(\sigma_B^L[0..2^n - 1]) = S_I$. So by the construction of $\sigma_C^L[0..2^n - 1]$, $|S_{max}^L \setminus F(\sigma_C^L[S_{max}^L])| \geq |S_I \setminus F(\sigma_B^L[S_I])| - 1$. Thus $n_i \geq |S_I \setminus F(\sigma_B^L[S_I])| - 1$.

Then recall that by the construction of $\sigma_B^L[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, $\sigma_B^L[S] = \perp$ if and only if $\sigma_A^L[S] = \perp$. By Lemma 14 and Lemma 12, Invariant (I2) holds for both $\sigma_B^L[0..2^n - 1]$ and $\sigma_A^L[0..2^n - 1]$, with $S_{max}(\sigma_B^L[0..2^n - 1]) = S_{max}(\sigma_A^L[0..2^n - 1]) = S_I$. So $|S_I \setminus F(\sigma_B^L[S_I])| = |S_I \setminus F(\sigma_A^L[S_I])|$. Thus $n_i \geq |S_I \setminus F(\sigma_A^L[S_I])| - 1$.

Next, recall that by the construction of $\sigma_A^L[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_I$, then $\sigma_A^L[S] = \perp$; otherwise $\sigma_A^L[S] = \sigma_B^S[S]$. By definition, $S_I = F(\sigma_B^S[S_{max}^S]) \cup I$.

Furthermore, by Lemma 10, $\sigma_B^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{max}(\sigma_B^S[0..2^n - 1]) = S_{max}^S$. So by Invariant (I4), $F(\sigma_B^S[S_{max}^S]) = F(\sigma_A^L[S_I])$. By construction, $I \cap F(\sigma_B^S[S_{max}^S]) = \emptyset$. Thus $n_i \geq |S_I \setminus F(\sigma_A^L[S_I])| - 1 = |I| - 1$.

By Lemma 11, $|I| \geq |L|/(7wk)$. Recall that in a low contention phase, $|L| \geq 0.5|S_{max}^S \setminus F(\sigma_B^S[S_{max}^S])|$. Thus $n_i \geq |L|/(7wk) - 1 \geq |S_{max}^S \setminus F(\sigma_B^S[S_{max}^S])|/(14wk) - 1$.

Case 2. $\sigma_{round}[i, 0..2^n - 1] = \sigma_C^G[0..2^n - 1]$.

Then by Lemma 20, $S_{max}(\sigma_C^G[0..2^n - 1]) = S_{max}^G$. Thus $n_i = |S_{max}^G \setminus F(\sigma_C^G[S_{max}^G])|$.

Now recall that in the construction of $\sigma_C^G[0..2^n - 1]$, we checked whether there exists a process $p \in S_G \setminus F(\sigma_B^G[S_G])$ such that p is within the critical section at the end of $E(\sigma_B^G[S_G])$. If such a process p exists, then we set $\sigma_C^G[0..2^n - 1]$ to be a simple modification of $\sigma_B^G[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains p has had $\sigma_C^G[S]$ set to \perp ; otherwise we set $\sigma_C^G[0..2^n - 1] = \sigma_B^G[0..2^n - 1]$. By [Lemma 19](#), $S_{\max}(\sigma_B^G[0..2^n - 1]) = S_G$. So by the construction of $\sigma_C^G[0..2^n - 1]$, $|S_{\max}^G \setminus F(\sigma_C^G[S_{\max}^G])| \geq |S_G \setminus F(\sigma_B^G[S_G])| - 1$. Thus $n_i \geq |S_G \setminus F(\sigma_B^G[S_G])| - 1$.

Then recall that by the construction of $\sigma_B^G[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, $\sigma_B^G[S] = \perp$ if and only if $\sigma_A^G[S] = \perp$. By [Lemma 17](#) and [Lemma 19](#), Invariant (I2) holds for both $\sigma_A^G[0..2^n - 1]$ and $\sigma_B^G[0..2^n - 1]$, with $S_{\max}(\sigma_B^G[0..2^n - 1]) = S_{\max}(\sigma_A^G[0..2^n - 1]) = S_G$. So $|S_G \setminus F(\sigma_B^G[S_G])| = |S_G \setminus F(\sigma_A^G[S_G])|$. Thus $n_i \geq |S_G \setminus F(\sigma_A^G[S_G])| - 1$.

Next, recall that by the construction of $\sigma_A^G[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_G$, then $\sigma_A^G[S] = \perp$; otherwise $\sigma_A^G[S] = \sigma_B^S[S]$. By definition, $S_G = F(\sigma_B^S[S_{\max}^S]) \cup S_Y$.

Furthermore, by [Lemma 10](#), $\sigma_B^S[0..2^n - 1]$ is $(i-1)$ -compliant with $S_{\max}(\sigma_B^S[0..2^n - 1]) = S_{\max}^S$. So by Invariant (I4), $F(\sigma_B^S[S_{\max}^S]) = F(\sigma_A^G[S_G])$. By definition, $S_Y \cap F(\sigma_B^S[S_{\max}^S]) = \emptyset$. Thus $n_i \geq |S_G \setminus F(\sigma_A^G[S_G])| - 1 = |S_Y| - 1$.

By [Lemma 16](#), $|S_Y| > \frac{|H|}{16k}$. Recall that in a high contention phase, $|H| > 0.5|S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])|$. Thus $n_i \geq \frac{|H|}{16k} - 1 \geq |S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])| / (32k) - 1$.

Case 3. $\sigma_{\text{round}}[i, 0..2^n - 1] = \sigma_C^H[0..2^n - 1]$.

Then by [Lemma 30](#), $S_{\max}(\sigma_C^H[0..2^n - 1]) = S_B$. Thus $n_i = |S_B \setminus F(\sigma_C^H[S_B])|$.

Recall that by definition, $S_B = S_\beta \cup F(\sigma_A^H[S_H])$. Furthermore, by [Lemma 30](#), $F(\sigma_C^H[S_B]) = F(\sigma_B^H[S_B]) \cup S_\alpha$.

Also recall that by the construction of $\sigma_B^H[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, if $S \not\subseteq S_B$, then $\sigma_B^H[S] = \perp$; otherwise $\sigma_B^H[S] = \sigma_A^H[S]$. Thus by [Lemma 21](#) and Invariant (I4), $F(\sigma_A^H[S_H]) = F(\sigma_B^H[S_B]) = F(\sigma_C^H[S_B])$.

Next, recall that by construction, $S_\alpha \cap F(\sigma_A^H[S_H]) = \emptyset$ and $S_\beta \cap F(\sigma_A^H[S_H]) = \emptyset$. Furthermore, by definition, $S_\beta = S_\alpha \cup \mathcal{Z}$. Therefore:

$$\begin{aligned} S_B \setminus F(\sigma_C^H[S_B]) &= S_B \setminus (F(\sigma_B^H[S_B]) \cup S_\alpha) \\ &= S_B \setminus (F(\sigma_A^H[S_H]) \cup S_\alpha) \\ &= (S_\beta \cup F(\sigma_A^H[S_H])) \setminus (F(\sigma_A^H[S_H]) \cup S_\alpha) \\ &= S_\beta \setminus S_\alpha &= \mathcal{Z} \end{aligned}$$

By definition, $\mathcal{Z} = \bigcup_{j \in I^D} z_j^D$. So $|\mathcal{Z}| = I^D$. Thus by [Lemma 22](#), $n_i = |\mathcal{Z}| \geq m/2$.

Now recall that:

- $m = 4|X|/k$.
- $|X| > |X'|$ and $\{X, X'\}$ is a partition of H_3 , so $|X| > |H_3|/2$.
- $|H_3| > |H|/32$.

So $n_i \geq m/2 > \frac{|H|}{32k}$.

Recall that in a high contention phase, $|H| > 0.5|S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])|$. Thus $n_i > \frac{|H|}{32k} > |S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])| / (64k)$.

So in all three cases, $n_i \geq |S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])| / (64wk) - 1$.

Now recall that in the construction of $\sigma_B^S[0..2^n - 1]$, we checked whether there exists a process $p \in S_{\max}^{\text{old}} \setminus F(\sigma_A^S[S_{\max}^{\text{old}}])$ such that p is within the critical section at the end of $E(\sigma_A^S[S_{\max}^{\text{old}}])$. If such a

process p exists, then we set $\sigma_B^S[0..2^n - 1]$ to be a simple modification of $\sigma_A^S[0..2^n - 1]$ where every set $S \subseteq \mathcal{P}$ that contains p has had $\sigma_B^S[S]$ set to \perp ; otherwise we set $\sigma_B^S[0..2^n - 1] = \sigma_A^S[0..2^n - 1]$. By Lemma 9, $S_{\max}(\sigma_A^S[0..2^n - 1]) = S_{\max}^{\text{old}}$. So by the construction of $\sigma_B^S[0..2^n - 1]$, $|S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])| \geq |S_{\max}^{\text{old}} \setminus F(\sigma_A^S[S_{\max}^{\text{old}}])| - 1$. Thus $n_i \geq |S_{\max}^S \setminus F(\sigma_B^S[S_{\max}^S])| / (64wk) - 1 \geq |S_{\max}^{\text{old}} \setminus F(\sigma_A^S[S_{\max}^{\text{old}}])| / (64wk) - 2$

Recall that by the construction of $\sigma_A^S[0..2^n - 1]$, for every set $S \subseteq \mathcal{P}$, $\sigma_A^S[S] = \perp$ if and only if $\sigma_{\text{old}}[S] = \perp$. By Lemma 9, Invariant (I2) holds for $\sigma_A^S[0..2^n - 1]$ and $S_{\max}(\sigma_A^S[0..2^n - 1]) = S_{\max}^{\text{old}}$. Also recall that by definition, $\sigma_{\text{old}}[0..2^n - 1]$ is $(i - 1)$ -compliant and $S_{\max}(\sigma_{\text{old}}[0..2^n - 1]) = S_{\max}^{\text{old}}$. So $|S_{\max}^{\text{old}} \setminus F(\sigma_A^S[S_{\max}^{\text{old}}])| = |S_{\max}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\max}^{\text{old}}])|$. Thus $n_i \geq |S_{\max}^{\text{old}} \setminus F(\sigma_A^S[S_{\max}^{\text{old}}])| / (64wk) - 2 \geq |S_{\max}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\max}^{\text{old}}])| / (64wk) - 2$

Finally, recall that $\sigma_{\text{old}}[0..2^n - 1]$ is simply $\sigma_{\text{round}}[i - 1, 0..2^n - 1]$. So $|S_{\max}^{\text{old}} \setminus F(\sigma_{\text{old}}[S_{\max}^{\text{old}}])| = n_{i-1}$. Consequently, $n_i \geq n_{i-1} / (64wk) - 2$. Then, since $k = w^d$, $n_i \geq n_{i-1} / (64w^{d+1}) - 2$.