Decomposition and infima in the computably enumerable degrees

Rodney G. Downey[*]
School of Mathematical and Computing Sciences
Victoria University of Wellington
Geoffrey L. LaForte
Department of Computer Science
University of West Florida
Richard A. Shore [**]
Department of Mathematics
Cornell University

Abstract

Given two incomparable c.e. Turing degrees $\mathbf{a}$ and $\mathbf{b}$, we show that there exists a c.e. degree $\mathbf{c}$ such that $\mathbf{c} = (\mathbf{a} \cup \mathbf{c}) \cap (\mathbf{b} \cup \mathbf{c})$, $\mathbf{a} \cup \mathbf{c} \big| \mathbf{b} \cup \mathbf{c}$, and $\mathbf{c} < \mathbf{a} \cup \mathbf{b}$.

1 Introduction

It is well-known that the partial order of the Turing degrees forms an upper semilattice. This fact arises because there is a natural notion of the sum of the information contained in a pair of sets, and this can be captured by merely representing the first set as a subset of the even numbers, and the second set as a subset of the odd numbers. On the other hand, there is no natural notion

[*]First author supported by Marsden Grant for Basic Science, New Zealand.
[**]Third author supported by NSF Grants DMS-9802843, DMS-0100035, and INT-9602579.

of the greatest information that two sets share, a fact first demonstrated explicitly by Kleene and Post [4], who constructed a pair of degrees with no common greatest lower bound. Although a pair of degrees in general may fail to have an infimum, the existence of degrees with infimum $\mathbf{0}$ (or any degree at all by relativization) can be established by the same methods. Returning to joins, however, it is not true that any degree is the supremum of a pair of degrees strictly below it, since there are minimal degrees that fail to have any nontrivial degrees below them at all, as demonstrated by Spector in [14]. Hence, for the partial order of all Turing degrees the situation with regard to infima and suprema is completely complementary: Every incomparable pair of degrees has a supremum, but not every degree has a decomposition into an incomparable pair below it. Every degree branches into a pair of incomparable degrees above it of which it is the infimum, but not every pair of incomparable degrees has an infimum.

In the more effective world consisting solely of the computably enumerable degrees, there is even more difference between the existence of joins and that of meets. It is easy to see that the join of any two c.e. degrees is still c.e. but in this case, the Sacks Splitting Theorem, [9], shows that in fact every nontrivial degree can be decomposed as the supremum of two low incomparable degrees. The existence of pairs of computably enumerable degrees with no infimum was established by Lachlan [5] and Yates [16]. Lachlan, however, also showed in [5] that not every incomplete c.e. degree is the infimum of two incomparable degrees.

Given this basic situation, it is natural to wonder how uniformly distributed these phenomena are in the partial order of the c.e. degrees. More precisely we can ask to what extent these general facts are reflected in nondegenerate intervals of the c.e. degrees. A straightforward modification of the Sacks Density Theorem [10] shows that between any two c.e. degrees must lie a pair of incomparable degrees. Hence, the notions of meet and join are not completely vacuous locally, and any such pair of incomparable degrees must of course have a supremum in the interval. On the other hand, Lachlan's Nonsplitting Theorem, [6], demonstrated the existence of an interval with a top which could not be decomposed as a join over its bottom. This shows that the local situation for joins can be very different from the global one.

For infima the situation inside nondegenerate intervals became clear somewhat later. First the density of c.e. degrees which are not infima was established by Fejer [3]. Then the density of pairs of degrees with no infimum was established by Ambos-Spies, [1]. This left open the question of whether every nonempty interval in the c.e. degrees contained a pair of incomparable degrees with an infimum until this was established by Slaman [13]. This shows that the situation for meets inside every nondegenerate interval of the c.e. degrees reflects the global situation exactly, where as that for joins does not, with the essential difference given by the Lachlan Nonsplitting Theorem.

Since any pair of degrees has a unique join, it is trivial to see that any pair of degrees with an infimum must have a join, and that this join remains inside any interval containing the pair. Not every decomposition of a c.e. degree, however produces a pair with an infimum, since there are c.e. degrees with no infimum. Lachlan, [8], however, did show that every degree does have a decomposition as the join of a pair of degrees with an infimum.

We extend this splitting result of Lachlan to intervals by showing that for any pair of degrees $\mathbf{a} < \mathbf{b}$, $\mathbf{b}$ can be split over $\mathbf{a}$ if and only if $\mathbf{b}$ can be split as a pair with infimum over $\mathbf{a}$. Thus the obstacle to decomposing a degree into a pair with an infimum inside a specific interval is due solely to the impossibility of decomposing it in any way at all in the interval. This follows from our main theorem which in some sense shows that every decomposition of a degree into a join of incomparables is close to a decomposition into a pair that has an infimum.

Theorem 2.1 *For any computably enumerable A, $B \subseteq \omega$, if $A|_T B$, then there exists a computably enumerable C such that C is the infimum of $A \oplus C$ and $B \oplus C$, $(A \oplus C)|_T(B \oplus C)$, and $C \leq_T A \oplus B$.*

This application of our result to the relationship between Sacks splitting and the Lachlan splitting in an interval also answers affirmatively an open question of Slaman and Shore mentioned in [12]. In this paper, they give a proof of the result of Harrington that every low$_2$ $\mathbf{b}$ can be split over any $\mathbf{a}$ below it, and then ask whether every such $\mathbf{b}$ can be split into a pair with infimum over every $\mathbf{a} < \mathbf{b}$. Of course, by our result no use of low$_2$-ness is needed here.

Not only can Lachlan's Splitting Theorem be derived from the simpler Sacks Splitting Theorem using our result, but we also note that the Slaman Density Theorem for infima follows from our result together with the existence of incomparable pairs in any nondegenerate interval.

Our proof has an added technical interest as an extension of the method of Shore, [11] to a more complex tree of strategies with branchings of width more than ω+1.

2 The theorem and the intuition for its proof

Theorem 2.1. *For any computably enumerable A, $B \subseteq \omega$, if $A|_T B$, then there exists a computably enumerable C such that C is the infimum of $A \oplus C$ and $B \oplus C$, $(A \oplus C)|_T(B \oplus C)$, and $C \leq_T A \oplus B$.*

Our proof involves a priority argument using an ω^2+1-branching tree of strategies. However, the nodes indexing strategies which actually act during the construction form a subtree of the full tree which is merely ω+1-branching. The reader should be familiar with the tree method of organizing strategies in priority arguments, for which chapter XIV of [15] is the standard reference. Our notation generally follows that found there. It might also be helpful to the reader to have some acquaintance with [2] and [11], where ω+1-branching trees are discussed in some detail.

Let A and B be incomparable c.e. sets. Letting Φ and Ψ range over all computable functionals, in order to construct C, we must satisfy the following indexed families of requirements:

M_Φ : $\forall X\big((\Phi(A \oplus C) = \Phi(B \oplus C) = X) \Rightarrow X \leq_T C\big)$,

N^A_Ψ : $\Psi(A \oplus C) \neq B$, and

N^B_Ψ : $\Psi(B \oplus C) \neq A$.

We will be able to show that $C \leq_T A \oplus B$ by analyzing the construction, so we do not introduce extra requirements for this purpose.

In isolation, the strategies for these requirements are simple. The infimum requirements of type M will be satisfied by a trace-setting procedure. At each stage at which some new number k appears such that $\Phi(A \oplus C) \upharpoonright k = \Phi(B \oplus C) \upharpoonright k$, we appoint a trace $c(k)$ targeted for C. If either A or B later changes below its use on the relevant computation, we will try to enumerate $c(k)$ into C unless there is some other strategy with authority to restrain us from doing so. When new computations for these values appear again, we pick a new larger trace and repeat this procedure. In this way, assuming the permanent restraint imposed on the traces for M is finitary, we can compute $\Phi(A \oplus C) \upharpoonright k = \Phi(B \oplus C) \upharpoonright k$ from C.

The incomparability requirements of type N will be satisfied by the Sacks preservation method. Consider a requirement of type N^A. (The strategies for type-N^B requirements are mirror images of the ones for requirements of type

N^A.) Whenever a new number x appears such that $\Psi(A \oplus C) \upharpoonright x = B \upharpoonright x$, we restrain C from changing below the use of this computation. If we do this for every number x, then we ensure that only some A-computable part of C is used to determine $\Psi(A \oplus C)$, hence some A-computable procedure produces the same function. Therefore, if $\Psi(A \oplus C)$ were to compute B, A would do so as well, contrary to hypothesis.

2.1 Simple interaction between strategies

The conflict between these two kinds of strategies is stark. Strategies for the positive type-M requirements are essentially aimed at making $C \equiv_T A \oplus B$, while those for the negative type-N requirements are aimed at making $C \equiv_T \emptyset$. More precisely, a conflict occurs whenever a negative strategy, for example some $\mathrm{N} = \mathrm{N}^A_\Psi$, needs to interfere with some higher-priority strategy for $\mathrm{M} = \mathrm{M}_\Phi$ by restraining some M-trace $c(k)$. The key to overcoming this basic obstacle, somewhat as in Slaman, [13], is to resolve conflicts by demanding that a lower-priority N-strategy take into account $\phi(A \oplus C; k)$ in every calculation of $\psi(A \oplus C; x)$ which appears after $c(k)$ is appointed. We can achieve this by replacing the ordinary use $\psi(A \oplus C; x)$ by an artificially increased $\psi^+(A \oplus C; x)$ for the sake of controlling the imposition of restraint by the N-strategy. In other words, the use on x of the reduction $B \leq_T A$ that the N strategy threatens to build incorporates $\phi(A \oplus C; k)$ as well as $\psi(A \oplus C; x)$. If $\Phi(A \oplus C; k)\downarrow$ this will have only finite effect on the N-strategy. If $\Phi(A \oplus C; k)\uparrow$, the N-strategy will eventually have an opportunity to impose restraint at a stage when $c(k)$ is undefined, and so avoid all future conflict.

The procedure thus outlined is a little too simple to ensure that all conflicts between the strategies for M and N can be resolved, since it only ensures that for each trace $c(k)$, the strategy for computing $\Phi(k)$ from C inflicts finite injury on the strategy for N. If the M-strategy succeeds in building a total functional, it will have infinitely many traces defined in the limit, so that the N-strategy cannot possibly take all of the associated computations into account. However, if the computation $\Psi(A \oplus C; x)$ actually converges, then there is no need to take into account any of the uses for computations associated with traces that are larger than $\psi(A \oplus C; x)$. Hence, there need only be a finite number of values of $\phi(A \oplus C)$ which must be taken into account when defining the increased $\psi^+(A \oplus C; x)$. This still enables M to be satisfied, since whenever the restraint drops and some $c(k)$ enters C, all greater $c(k')$ will also enter and become undefined. If this entry of $c(k)$ has no effect on $\Psi(A \oplus C; x)$, then all new traces for the M-strategy will be chosen larger than the N-strategy's restraint, and so will never be involved in $\psi^+(A \oplus C; x)$ again. If the entry of $c(k)$ causes $\Psi(A \oplus C; x)\uparrow$, then we have authority to increase ψ^+ again anyway, since the original computation has been destroyed.

Although this procedure yields a solution to the basic conflict between the two kinds of requirements, the fact that each strategy for an infimum requirement of type M must succeed in the presence of many strategies for lower-priority requirements of type N introduces a more significant problem. Consider the situation where $\mathrm{M} = \mathrm{M}_\Phi$ has higher priority than two negative requirements $\mathrm{N}^A = \mathrm{N}^A_{\Psi_0}$ and $\mathrm{N}^B = \mathrm{N}^B_{\Psi_1}$, where, say, N^B has higher priority than N^A. Suppose we have set a trace $c(k)$ based on initial computations for $\Phi(A \oplus C; k)$ and $\Phi(B \oplus C; k)$. First suppose the N^A-strategy imposes a restraint for the sake of some x at a stage s_0 which would keep $c(k)$ from entering C. At stage $s_1 > s_0$, some change occurs in B below $\phi(B \oplus C; k)$. A has not changed, so we cannot enumerate $c(k)$ into C without injuring the N^A-strategy. Next, at stage $s_2 > s_1$, the N^B-strategy imposes restraint for the sake of some other number x'. Then, at stage $s_3 > s_2$, A changes on $\phi(A \oplus C; k)$, dropping its prohibition on $c(k)$ entering C. But now the N^B-strategy's restraint has been imposed, and $c(k)$ is prevented from entering C. The problem is, as usual, that even though we have waited for the computation at k to recover and so both sides still give the original answer, some larger number may now go into C destroying both sides of the computation without our being able to put $c(k)$ into C. We must let N^A and N^B impose additional restraint (as mentioned above) to prevent any action (i.e. putting numbers into A or B) that might that make us want to put these larger numbers into C. The problem is how to do this without letting the restraint imposed become unbounded.

The restraint imposed by a negative strategy such as N^B is in danger of becoming unbounded simply because a lower priority type-N^A strategy may be holding the trace $c(k)$ out of C when the use of $\phi(B \oplus C; k)$ increases. In this case, the increase in the restraint that the strategy for N^B can further restrain a still larger trace, for example $c(k+1)$ which is also held out by some lower-priority strategy. This pattern can repeat itself as more and more traces are defined tied to computations that stabilize at later and later stages. In other words instability of $\phi(k+1)$ causes added restraint on $c(k+2)$, then the $k+2$ computation affects $c(k+3)$, and so on. Suppose the N^B strategy involves the functional Ψ, that is, it is a strategy for N^B_Ψ. Then the increased use $\psi^+(B\oplus C;x)$ could go to infinity for some fixed n because it is now forced to take into account more and more traces, although $\psi(B\oplus C;x)$ itself has never changed.

There are two ways in which we avoid this problem: first, lower priority strategies (like that for N^A here) turn out to be prevented from reimposing restraint once they have released and some higher priority requirement such as N^B has taken over until the higher priority one releases. This, however, is not quite enough, since completely new restraints could still be set by lower-priority requirements that have never yet held $c(k)$ out of C. To avoid this, we distinguish between the situations where ψ^+ increases "honestly", that is, through an increase in ψ itself, and those in which the increase is caused by some value of $\phi(B\oplus C;k)$ which the N^B_Ψ-strategy believes will converge in the limit. Essentially, this means allowing only those strategies believing that $\psi^+(x)\!\downarrow$ to act as long as $\psi(x)$ is stable, but also initializing all such strategies whenever $\psi^+(x)$ changes. In this way, such strategies can be prevented from holding $c(k)$ when this would interfere with the N^B_Ψ strategy in this way. If the N^B_Ψ-strategy has the correct guess about the higher-priority type-M requirements which it must respect, $\psi^+(x)$ will only change finitely often after $\psi(x)$ stabilizes. Thus this extra initialization can only impose finite injury below the correct outcome of the N^B_Ψ strategy.

2.2 The development of the priority arrangement

So far we have not been explicit about how a guess about the ultimate outcome of a strategy for a type-M requirement should be implemented in our construction. The way that a requirement M_Φ is satisfied depends directly on a Π^0_3 condition — whether or not $\Phi(A\oplus C)=\Phi(B\oplus C)$. As we approximate the truth of this condition throughout the construction, a strategy for M_Φ appoints more and more traces, each of which must be taken into account by lower-priority strategies for negative requirements. Hence, a negative requirement that guesses that the Π^0_3 condition is true (the ∞ outcome) guesses that the functional the infimum strategy is building is total and therefore expects to take into account infinitely many traces appointed by this strategy. One that guesses that the condition is false, on the other hand, expects there to be some k at which the two functionals of the requirement fail to be defined and equal for infinitely many stages s. Such a strategy will be able to ignore almost all of the M-strategy's activity by acting only at stages when $\Phi(A\oplus C)$ and $\Phi(B\oplus C)$ fail to converge to the same value at k. As discussed in detail in Downey [2], such a situation suggests a sequence of Π^0_2 outcomes, one for each k that can serve as a witness to the failure of the Π^0_3 condition of the requirement. Since the truth of this Π^0_3 condition is equivalent to the failure of all of these Π^0_2 conditions, the natural framework to use for our proof is the ω+1-branching tree method of Shore [11].

We begin with a base tree of strategies that is (isomorphic to) a subtree of ${}^{<\omega}(\omega^2{+}1)$. As we explain below, our actual tree of strategies will be the limit of a sequence of subtrees of ${}^{<\omega}(\omega^2{+}1)$ which we define during the construction. We order the outcomes of a strategy for a (higher-priority) requirement $\mathrm{M} = \mathrm{M}_\Phi$ in type $\omega^2{+}1$ as a natural way of encoding enough information about the strategy's outcome for lower-priority requirements. We use $\omega \times \omega$ with lexicographic order for the set of outcomes below ω^2. (In other words, we write $\omega{\cdot}k + l$ as $\langle k, l\rangle$.)

The outcome $\langle k, l\rangle$ is the leftmost one accessible infinitely often if and only if k is least such that $\neg[\Phi(A \oplus C; k)\downarrow= \Phi(B \oplus C; k)]$ and l is the least such that $c(l)$ is undefined infinitely often.

The additional infinitary outcome to the far right, ∞ (or ω^2), will be the leftmost accessible infinitely often if the lim sup of the length of agreement between $\Phi(A{\oplus}C)$ and $\Phi(B{\oplus}C)$ is infinite. We will implement our strategies in such a way that it will be possible to show below that for each k there is some least l such that no outcome $\langle k, l'\rangle$ for $l' \geq l$ is ever accessible in the construction. Thus our construction actually takes place on a ω+1-branching subtree of our priority tree. If $\Phi(A \oplus C) = \Phi(B \oplus C)$, then the infinite outcome to the far right will lie on the true path for the construction, and the strategies below this point will have to be able to coordinate their actions somehow with the strategies extending finitary outcomes of the strategy for M.

The dependence of lower-priority N-strategies on the eventual outcome of M is more delicate than that which arises in the analogous situation in [11]. There, the sets being built were K-REA, and all changes affecting approximations in the construction resulted directly from enumerations into K. This meant that even though a return at a later stage to a Π^0_2 outcome already visited at a previous stage did not initialize all of the strategies extending the Π^0_3 outcome lying to the right, no actions taken by such strategies during the interval between any such pair of stages could leave any trace in the sets being constructed. Here, where any enumeration into the truly c.e. set C is permanent, the situation is completely different. The infinitely many strategies lying to the left of strategies below an infinitary outcome to the far right need to know what effect, in terms of both restraint-setting and trace-enumeration, these higher-priority strategies will have on the construction.

We are therefore forced to equip each strategy with a guess as to the outcome of higher priority strategies lying to its right. We implement the extra guessing involved in a dynamic fashion, changing the original priority ordering by adding new nodes encoding more information whenever the troublesome situation presented by nodes which have to guess at the infinitary activity of an infimum-preserving strategy to their right appears. Although it is possible to calculate a static arrangement in advance which assigns priorities to strategies correctly, the dynamic assignment is more in keeping with our intuition for the construction. Additionally, a static arrangement makes the already complex definition of the construction even more difficult to understand.

We change the priority of the strategies each time this infinitary outcome is visited. Whenever a new strategy extending this rightmost outcome first wishes to act, we give it the authority to interfere with all strategies for requirements below outcomes of the M-strategy which are to the right of some outcome not yet visited.

Our procedure introduces some technical problems that produce considerable complexity in the full construction. In order to give the reader enough intuition to follow the details in section 3 below, we sketch the main ideas involved now. Each strategy α extending the infinite outcome of an M-strategy will be tied to a number k^{α}, its *killing point* on the M-strategy. This will be some number for which no trace has yet been appointed by the M-strategy before the stage at which α first acts. If the M-strategy's computation on any $k \leq k^{\alpha}$ ever changes thereafter, α will be initialized. At the stage at which α first acts we change the tree of strategies by inserting a *ghost* of α before every strategy extending any $\langle k, l \rangle$ for $k > k^{\alpha}$. This changes the implicit priority arrangement by giving α the authority to interfere with all such strategies. The most serious problem remaining for the construction is to determine what the outcomes of α's ghosts should be. This presents a significant obstacle, since the nodes which lie to the right of the true path may act many times in the intervals between the stages at which the true path is accessible.

For a node associated with a negative incomparability strategy, there is no serious difficulty. The natural choice is the least index of a restraint that has diverged at some point during the interval. There is a little subtlety involved here depending on whether or not the divergence of the restraint was honestly due to the negative strategy itself or to the effect of some higher-priority type-M strategy. Since a ghost of α encodes different guesses from those of α about which computations are eventually destined to converge, its outcome needs to reflect this information by using it to decide whether or not the restraint involved is destined to eventually remain permanent. The details for achieving this are given in 3.6 below.

For a node associated with a positive infimum strategy, the situation is apparently more troublesome. At any finite stage there are only finitely many traces assigned, but, of course, the $\liminf$ of the trace-assigning procedure could still be infinity. If this is the case, the outcome of the ghost should apparently be ∞ like the outcome of the actual node. Unfortunately, allowing infinitary branching like this to be copied back into the tree threatens the well-foundedness of our whole procedure. This is because we might be forced to have an infinite sequence of ghost outcomes with no actual strategy intervening on the true path. This would happen if we were to allow each activity below an ω^2 outcome to introduce ω^2+1 sequences of ghosts, like a hydra that grows back infinitely many heads after each one is chopped off. Although this looks worrisome, we are saved by the fact that all but finitely much activity below any node to the right of the true path is initialized infinitely often. Since this applies to the traces belonging to nodes with ghosts on the true path, each such node will have some greatest trace that is permanent, and the ghost node need never get an infinitary rightmost outcome. Hence a type-M node can be treated much as a type-N node is. The ghost of the higher-priority node gets as outcome $\langle k, l\rangle$, where k is least such that the functionals involved in the requirement have failed to converge equal to each other on k, and l is least such that $c(l)$ has been undefined at some stage in the interval since the ghost was last visited.

This assignment of outcomes for ghosts ensures that the sequence of outcomes accessible at the ghost has the same $\liminf$ as the sequence at the actual node. However, a problem arises from the fact that the current state of some strategy at stage s may be different from the outcome of its ghost included in the nodes which act at stage s. In the case of type-N requirements, this means that some restraint may be set at every stage when nodes on the true path act, even though this restraint is removed infinitely often. Thus some currently-restrained traces might be allowed to enter C later, thereby injuring the strategies of nodes on the true path. In the case of type M requirements, the analogous problem arises when some trace $c(k)$ exists at every stage when nodes on the true path act, even though this trace is undefined infinitely often. Thus nodes to the right still threaten to injure strategies of lower priority to the left, even though these lower-priority strategies have a truthful ghost on which to rely. Fortunately, any negative strategy to the right eventually has to give up authority over each particular trace with unstable restraint, and any infimum strategy must pick larger and larger numbers in its attempts to set a trace for a computation that is undefined in the limit. Thus, nodes on the true path can afford to delay the imposition of their restraints until all traces which might interfere with their strategies enter C. This is the reason for the definition of interference in section 3.2 below.

Our procedure of introducing ghost outcomes means that we have to initialize nodes to the left of the most recent approximation to the true path whenever the outcome they were depending on is shown to be false. This procedure is straightforward, and natural, although a little tedious to define.

A more important issue is that our construction results in two different "priority" relations between strategies: the $<_L$ relation on the tree which is the limit of the process of adding more nodes, and the other is the relation of being first accessible at an earlier stage in the construction. Each of these relations is the basis for a kind of priority among the strategies, and the actions taken during the construction depend on both. They must be distinguished, since their union is not a linear ordering and so cannot lead to a single notion of priority in the ordinary sense. In most respects, the latter relation (of prior first action) is the more important one, although it is derived from the former relation.

The fact that we have two different kinds of priority appears to cause a problem when two extensions of an ∞ outcome have killing points assigned in the opposite order from their natural priority arrangement, particularly in the assignment of ghost outcomes. For example, suppose $\beta_1 <_L \beta_2$, $\alpha^\frown\infty \subseteq \beta_1 \cap \beta_2$, and β_2 acted at a stage before any β_1-stage. Since β_1 must respect more strategies than β_2, it is hard to see how all conflicts can be avoided in this situation without a more detailed analysis of the possibilities. Notice that nodes to the left of the true path fall into two classes relative to each node β on the true path, depending on whether or not they include a ghost of β. There are finitely many that do not include such a ghost, and all of them stop acting forever after some point. Those that do include such a ghost, act almost exactly as though they were extensions of the actual strategy β. In either case, actions arising from strategies left of the true path eventually cease to have any effect on an individual strategy β on the true path. If

β_2 is on the true path then there is certainly nothing to worry about, since both β_1 and any other strategies which β_2 must worry about will eventually stop acting. If β_1 is on the true path, then, again, activity arising from the left of β_1 will eventually cease to have any effect. In this case, either activity involving $\beta_1 \cap \beta_2$ will initialize β_2 infinitely often, or $\beta_1 \cap \beta_2{}^\frown\infty \subset \beta_2$ with the killing point for β_2 below the outcome β_1 extends. In that case, we will arrange things so that all but some fixed finite part of β_2's activity is initialized infinitely often (by $\beta_1 \cap \beta_2$ activity) just before stages at which β_1 acts. Hence, as long as at least one of the two nodes lies on the true path, the other's activity will not cause permanent harm. If, on the other hand, the true path lies left of β_1, but right of β_1's killing point on α, then both β_1 and β_2 will have ghost outcomes along the true path; in this case, if β_1 acts infinitely often, the ghost outcome for β_2 will indicate that it is initialized infinitely often by β_1. If the true path lies between the killing points, then β_1 will be initialized infinitely often by the true path, but β_2 will not be affected by this initialization. A node on the true path will have ghost outcomes for both β_1 and β_2, and will be able to coordinate its strategy by waiting for all interference to go away.

3 Construction

The details of our construction are complex enough to be divided into six parts. First we define the sequence of trees which we use to produce the dynamic priority ordering. Second, we introduce some definitions and conventions which are used subsequently in describing the actions which take place in the construction. We then define the conditions under which elements are enumerated into C. Next we explain how the two types of strategies work. Finally, we define the ghost outcomes of the dummy nodes which code residual information from the right of the current approximation to the true path. We include a few intuitive remarks as we give the formal details to justify the complexity as much as possible.

3.1 The priority trees

Most of our notation is standard, as in [15], XIV. If α has requirement M_Φ assigned to it, we will often write Φ^α for Φ and ϕ^α for the associated use functional. If α has N_Ψ assigned to it, we will write Ψ^α and ψ^α for Ψ and its use functional. This makes it easier to refer to the functionals associated with several different strategies when we have to describe some action or condition involving all of them. In the case of all functionals, we may leave out explicit mention of the oracle if this can be done without confusion in the context. For example, if β is a type-N^A node, then $\phi^\beta(n)$ will mean $\phi^\beta(A;n)$. For technical convenience, we assume that all uses of computations are nondecreasing in the stage, increasing in the argument, and greater than 0 when their computations are convergent. If divergent we let the use be 0. Using functions giving the standard codings for n-tuples of natural numbers and indices of computable functionals and computably enumerable sets, we order each of the three types of requirements of section 2 in a priority listing. As discussed above, our construction will be controlled by a sequence of subtrees of ${}^{<\omega}(\omega^2{+}1)$, which we refer to as *priority trees.* As shown below in Lemma 4.3, once a node has been accessible, it will never be removed from subsequent trees. This approach is convenient for describing how we assign requirements to strategies, but it may give a better intuition to think of our construction as taking place on the gradually growing tree that is the limit of this sequence.

We will assign type-M requirements to even levels of our base priority tree, and alternate the assignment of type-N requirements to odd levels. As the trees involved become more complex, we will reassign requirements to nodes in a way consistent with this growing complexity. We begin at stage 0 with a base priority tree $\mathcal{T}[0]$ which is the subtree of ${}^{<\omega}(\omega^2{+}1)$ isomorphic to the subtree obtained by allowing only ω-branching at the odd levels. We write $\langle k,l\rangle$ for $\omega{\cdot}k{+}l$ and ∞ for ω^2 when discussing outcomes on the tree. We assign requirement M_n to each $\alpha \in \mathcal{T}[0]$ of length $2n$. We assign requirement N_n^A to each $\alpha \in \mathcal{T}[0]$ of length $4n+1$. We assign requirement N_n^B to each $\alpha \in \mathcal{T}[0]$ of length $4n+3$. Even-length nodes have outcomes $\langle k,l\rangle$ for each $k,l \in \omega$ and ∞, the ω^2 outcome. An outcome $\langle k,l\rangle$ indicates that least number on which the functional built by the strategy fails to converge is k, and that every trace assigned to computations on numbers greater than or equal to l eventually enters C. The oucome ∞ indicates that the functional built by the strategy is total. We say these nodes are of type M. Odd-length nodes have outcomes k for each $k \in \omega$. The outcome k indicates the least point of disagreement for the the functional and set involved in the associated requirement. We say these nodes are of type N.

In order for the strategies left of infinitary outcomes of type-M nodes to take into account the behavior of higher-priority nodes to their right, we must insert ghost-outcomes for these nodes below such strategies. We define the sequence of trees to which our ghost-inserting procedure gives rise by recursion, after first making some preliminary definitions. It is an essential feature of our construction that not every node on a priority tree has a requirement assigned to it for which it has responsibility. If β does have a requirement assigned to it we call β an *actual node.*

Suppose $\mathcal{T}$ is a priority tree, and α and β lie on $\mathcal{T}$, with $\beta <_L \alpha$. Let $\gamma = \beta \cap \alpha$. If γ is an actual type-M node and there are $k, l \in \omega$ such that $\gamma^\frown\langle k, l\rangle \subseteq \beta$ and $\gamma^\frown\infty \subseteq \alpha$, then we say α and β *have infinitary conflict on* $\mathcal{T}$.

As discussed above in section 2.2, this is the situation that makes ghost-inserting necessary. Along with our sequence of priority trees, we therefore define sets of ghost outcomes. For $\mathcal{T}[0]$, the set of *ghost outcomes on* $\mathcal{T}[0]$ is empty.

At stage $s+1$, we define the priority tree $\mathcal{T}[s+1]$. There will be at most one node on $\mathcal{T}[s]$ which is accessible for the first time at stage s. Recall from section 2.2 that we have an additional notion of priority between nodes that is generally more important than the standard notion for trees, especially when the nodes in question have infinitary conflict.

Definition 3.1. *Suppose α and β both lie on the same priority tree. Then α is* prior to *β if and only if the stage at which α was first accessible is less than the stage at which β was first accessible. β is* subsequent to *α if and only if α is prior to β.*

Note that at most one node will be accessible for the first time at any stage s so that the "prior-to" relation is a total ordering. If no node is accessible for the first time at stage s, then $\mathcal{T}[s+1] = \mathcal{T}[s]$. Otherwise, let α be the node first accessible at s and let $\gamma_0 \subset \gamma_1 \subset \ldots \subset \gamma_m \subset \alpha$ be the sequence of nodes γ such that $\gamma^\frown\infty \subseteq \alpha$. Note that any such γ is of type M. For each $i \leq m$, let $k_i = s$. Notice that for all $t \leq s$, and all $k \geq k_i$, $c^{\gamma_i}(k)\!\uparrow[t]$. At any stage when γ_i has outcome less than or equal to k_i, we intend to completely restart the α-strategy; we therefore say k_i is the *killing point of α on γ_i*, and we write $k^\alpha(\gamma_i)$ for it.

For each $i \leq m$, we must insert ghost outcomes for α into all nodes extending a finitary outcome of γ_i greater than α's killing point $k^\alpha(\gamma_i)$. Fix $i, k, l \in \omega$, with $i \leq m$ and $k > k^\alpha(\gamma_i)$, and let β_0 be a node of minimal length such that β_0 is not a ghost and $\gamma_i^\frown\langle k, l\rangle \subseteq \beta_0$. For each $\beta \in \mathcal{T}[s]$ such that $\beta_0 \subseteq \beta$, let δ^β be such that $\beta = \beta_0^\frown\delta$. (Of course, if $\beta = \beta_0$, $\delta^\beta = \lambda$.) If α

has type-N, we let, for each $n \in \omega$, $\beta_0^\frown n^\frown \delta^\beta \in \mathcal{T}[s+1]$. If α has type-M, we let, for each $x, y \in \omega$, $\beta_0^\frown \langle x, y\rangle^\frown \delta^\beta \in \mathcal{T}[s+1]$. We say β_0 is a *ghost of* α *on* $\mathcal{T}[s+1]$. With the exception of β_0 itself, each node $\beta_0^\frown n^\frown \delta^\beta$, or $\beta_0^\frown \langle x, y\rangle^\frown \delta^\beta$ is said to be a *new version* of β on $\mathcal{T}[s+1]$, has the same requirement (if any) assigned to it, and is of the same type as β. If $\beta \neq \beta_0$, then these new versions of β are said to have *ghost outcomes* on $\mathcal{T}[s+1]$ if and only if β had a ghost outcome on $\mathcal{T}[s]$; additionally, all outcomes of β_0 on $\mathcal{T}[s+1]$ are said to be *ghost outcomes*. A node is said to be a *ghost* if it has a ghost outcome. Note that the ghosts of type-M nodes have outcomes ordered in type ω^2, rather than ω^2+1, and that β_0 was not a ghost on $\mathcal{T}[s]$, but is one on $\mathcal{T}[s+1]$. Note also that any node β with a ghost inserted into it at s can never have been accessible before s, so that α is prior to any such β.

We allow ourselves a small notational license worth noting explicitly. Although the ghost outcomes of nodes have the same written forms (n and $\langle k, l\rangle$) as actual outcomes, we distinguish them from such outcomes. Thus an expression like '$\beta^\frown n$' can actually have two different meanings depending on whether β is a ghost or not. For the sake of complete precision, we should have introduced two disjoint sets of outcomes, one with subscripts to indicate ghost status, and we should have written expressions like $\beta^\frown n_g$. Because the tree on which we are considering such a β is eventually fixed, it should always be clear from the context which kind of node and outcome is under discussion. We chose to avoid making this distinction explicit in order to simplify our notation, but the difference must be kept in mind in several places below, for instance, in the discussion of the true path at the beginning of section 4.

Continuing our consideration of the modifications to the tree involving the node α that was accessible for the first time at stage s, if $\beta \in \mathcal{T}[s]$, and either $\alpha <_L \beta$, $\beta \subset \alpha$, or $\alpha \subseteq \beta$, then $\beta \in \mathcal{T}[s+1]$. If $\beta \in \mathcal{T}[s]$, $\beta <_L \alpha$, and β does not have an infinitary conflict with α on $\mathcal{T}[s]$, then $\beta \in \mathcal{T}[s+1]$. Hence, nodes that are already to the right of α or have no infinitary conflict with α do not get new versions on $\mathcal{T}[s+1]$.

For all $\beta \in \mathcal{T}[s+1]$, if β has a requirement assigned to it, we call any immediate outcome of β an *actual* outcome; recall that in this case we call β an actual node. Each actual type-N node β has restraint functions $r^\beta(y)[s]$ for each $y \in \omega$, and a special augmented use function, defined below and written ψ^+, when β has requirement N^A_Ψ or N^B_Ψ. Each actual type-M node β also has partial trace functions $c^\beta(k)[s]$ for each $k \in \omega$. We *initialize* a node β at stage $s+1$ by undefining all of its associated traces, functions and functionals.

In the construction, we will distinguish two kinds of stages: *destructive* stages at which previous work is undone, either by previously-set restraints becoming undefined or by some change in status of a previously-defined trace; and *constructive* stages at which new restraints can be set and traces can be defined. This is important because we must ensure that any required enumeration into C takes place before we set new restraints that could prevent such enumeration. At destructive stages, the current state of A and B directly determines changes in restraints and traces regardless of where the nodes to which they belong lie on the tree, as described in section 3.2 below. At constructive stages $s+1$, we will recursively define an *approximation to the true path*, $g_s \subset \mathcal{T}[s+1]$, and allow the strategies indexed by the initial segments of g_s to act at this stage, beginning with λ. For each $\alpha \subseteq g_s$, we say α is *accessible* at $s+1$ and call s an α-stage.

3.2 Restraints and traces

At any stage there will be some set of traces c that already wishes to enter C because a computation associated with c when it was first appointed has changed but is currently restrained by some negative strategy. An essential problem for the construction is to ensure that the a strategy for a negative requirement restrains only those traces which it really needs to, since restraining each such trace forces the strategy to take on responsibility for more and more computations. The definitions describing which nodes can restrain which traces are therefore basic to the entire construction, so we give them first in this section.

Definition 3.2. *We say a node $\alpha \in \mathcal{T}[s+1]$ of type N has* authority to restrain $c = c^{\beta}(k)[s]$ at s *if and only if there exist stages $s_0 \leq s_1 < s$ such that s_0 was an α-stage; $c^{\beta}(k)\uparrow[s_1]$; for every t with $s_0 \leq t \leq s$, α was not initialized at t; and either*

a. $\beta^\frown\infty \subseteq \alpha$,

b. there exists a ghost α_0 of α such that $\alpha_0 \subset \beta$, or

c. $\alpha \subset \beta$.

The definition above is intended to make our restraint-setting as weak as possible: we only allow a node α to take control of traces which were chosen after α's current strategy first began its activity, that is, only over traces which first converged with their current values at stages after the first α-stage since α was last initialized. Ensuring that a node that has never acted before cannot interfere with the already pending activity of other strategies makes it possible to compute the total restraint on a trace in such a way as to guarantee $C \leq_T A \oplus B$, since in this case only a finite number of strategies can ever restrain a trace from entering C.

We must allow type-N nodes authority to restrain type-M nodes with infinite outcome above them, since otherwise type-M nodes will make it impossible for strategies below their infinitary outcomes to satisfy their requirements. Notice that in cases (b.) and (c.) above, α cannot be expected to worry about the computations of such a type-M β below it, as there are infinitely many such β. Therefore, there is no notion that α must take such β into account when considering the conditions for the dropping of its restraint in Definition 3.4 below.

As discussed in section 2.2, because the outcome of a ghost must converge (in the sense of liminf) to the liminf of the outcome of the node of which it is the representative, the current state of the construction can be inconsistent with the information encoded in α by ghost outcomes. When this occurs, a node α's ability to set restraints is interfered with by the nodes about which it has incorrect information. This situation can also interfere with a type-M node's belief in a computation's stability. A type N-strategy causes interference when it is the most important one (in the sense of the prior-to relation) that restrains a trace that already wishes to enter C, but appears as a ghost which intends to give up this restraint eventually. Recall that each finitary outcome of a type-M node encodes two pieces of information — the least point of divergence and the least trace undefined infinitely often. Because of this, there are two subcases in the definition below to reflect the two different ways the current information could be false. Although the formal definitions will come later, we use $r^{\beta}(y)[s]$ to denote the restraint imposed by β at stage s for a computation at y and $r^{\beta}[s]$ to denote the maximum of all such restraints for β at s. We say that β *restrains c at s* if $r^{\beta}[s] > c$.

Definition 3.3. *Let $\alpha \in \mathcal{T}$ and $u \in \omega$, and suppose that either*

(1.) there exist y, z, a current trace $c < u$ that wishes to enter C at s, and a type-N $\beta \in \mathcal{T}[s]$ such that

a. β has authority to restrain c at s,

b. no node prior to β restrains c at s,

c. z is least such that $r^\beta(z)[s] > c$, and

d. the outcome of some ghost of β on α is $n \le z$; or

(2.) there exist y, a type-M $\beta \in \mathcal{T}[s]$, and a current k-trace for β, $c^\beta(k)[s] < u$ such that

a. $c^\beta(k)[s]$ does not wish to enter C at s and the outcome of the ghost of β on α is $\langle l, m\rangle$ for some $l \le k$, or

b. the outcome of the ghost of β on α is $\langle l, m\rangle$ for some $m \le k$.

If any of (1), (2a), or (2b), holds we say β interferes with u for α at $s+1$.

If β is any type-M node, we write Φ^β for the functional of β's requirement. (In other words, β has M_{Φ^β} assigned to it.) Similarly for a node α with N^A_Ψ or N^B_Ψ assigned to it we write Ψ^α for Ψ.

Suppose α is a type-N node with a requirement that involves the functional Ψ. In other words, α's requirement is either N^A_Ψ or N^B_Ψ. As described in section 2.1, strategies for negative requirements that set restraints must take responsibility for the computations of higher-priority infimum strategies that these restraints affect. We make the following recursive definition of the computations that a negative requirement must attempt to protect when it needs to set a restraint.

Definition 3.4. *Suppose α is assigned requirement N^A_Ψ at s. We say α* must respect $\phi^\beta(A \oplus C; z)$ for $\psi(A \oplus C; y)$ *at s if and only if $\beta^\frown\infty \subseteq \alpha$, α has authority to restrain $c^\beta(z)[s]$ at s, and either*

(1) $(c^\beta(z) < \psi(A \oplus C; y))[s]$, or

(2) there exists some z' and β' such that $\langle z', \beta'\rangle \ne \langle z, \beta\rangle$, α must respect $\phi^{\beta'}(A \oplus C; z')$ for $\psi(A \oplus C; y)$ at s, and $c^\beta(z) < \phi^{\beta'}(A \oplus C; z')[s]$.

We can now define an augmented use functional, ψ^+, for α's functional Ψ. We only give the definition for $\psi^+(A \oplus C; y)[s]$, since the version with $B \oplus C$ is the same with B replacing A everywhere. Note that when discussing α's activity at a stage s, we will never wish to count any computation of ψ^+

as convergent if some β interferes with such a computation for α at stage s. Hence, we let

$$\begin{aligned}&\psi^+(A \oplus C; y)[s] = \max(\{\psi(A \oplus C; y)[s]\} \cup \\ &\{\, \phi^\beta(A \oplus C; z)[s] \,:\, \alpha \text{ must respect } \phi^\beta(A \oplus C; z) \text{ for } \psi(A \oplus C; y) \text{ at } s \,\}),\end{aligned}$$

if no $\gamma \in \mathcal{T}$ interferes with $\psi^+(A \oplus C; y)[s]$ for α; otherwise, we let $\psi^+(A \oplus C; y)\!\uparrow\![s]$.

3.3 Trace enumeration

Our highest priority is achieving something for which we never introduced explicit requirements: we must be able to compute C from $A \oplus B$. Therefore, before calculating our approximation to the true path, we first allow all traces that wish to enter C at stage s the opportunity to do so. If we only let traces enter C when the nodes that appointed them are accessible at a stage, then we would need to calculate the true path in order to compute C. In this case, we could never hope to keep $C \leq_T A \oplus B$, since it requires both A' and B' to determine whether more expansionary stages for α will occur. As mentioned briefly before, our solution is to have a sequence of destructive stages occur during which all possible enumeration takes place before any constructive stage can take place at which nodes become eligible to act by being accessible when a new approximation to the true path is calculated. In this section we describe the activity that takes place during a destructive stage, while in the following sections we describe the actions for a constructive stage and the calculation of the approximation to the true path.

Because the assignment of priorities to strategies is more complex than in typical $\leq\omega$-branching priority constructions, it is convenient to introduce a general definition describing when activity involving a strategy forces the initialization of other strategies.

Definition 3.5. *Let $\alpha, \beta \in \mathcal{T}[s]$. For every outcome $\mathcal{O}$ of α,* the initialization *of $\alpha^\frown\mathcal{O}$* initializes *β* at *$s + 1$ if either*

(1.) α and β have no infinitary conflict on $\mathcal{T}[s]$, and $\alpha^\frown\mathcal{O} \leq_L \beta$;

(2.) α has type M, $\mathcal{O} = \langle k, l\rangle$ for some $k, l \in \omega$, and $\alpha^\frown\langle k', l'\rangle \subseteq \beta$ for some $k' \in \omega$ and $l' \geq l$;

(3.) $\alpha^\frown\infty \subseteq \beta$, $\mathcal{O} = \langle k, l\rangle$ *for some* $k, l \in \omega$, *and* $k \leq k^\beta(\alpha)$;

(4.) $\gamma = \alpha \cap \beta$, $\gamma^\frown\infty \subseteq \beta$, $\gamma^\frown\langle k, l\rangle \subseteq \alpha$ *for some* $k, l \in \omega$, *and* $k \leq k^\beta(\gamma)$;

(5.) there exists some ghost α_g *of* α, α_g *and* β *have no infinitary conflict on* $\mathcal{T}[s]$, *and* $\alpha_g^\frown\mathcal{O} \leq_L \beta$; *or*

(6.) there exists some ghost α_g *of* α, α *has type* M, $\mathcal{O} = \langle k, l\rangle$ *for some* $k, l \in \omega$, $\alpha_g^\frown\langle k', l'\rangle \subseteq \beta$ *for some* $k' \in \omega$ *and* $l' \geq l$.

We must add (2.) and (6.) above because the outcome $\langle k, l\rangle$ is caused by the enumeration of the trace $c(l)$, an event which can take place whether or not the computation tied to $c(k)$ is stable. Hence, we initialize $\langle k', l'\rangle$ even for $k' < k$ as long as $l \leq l'$. (3.) and (4.) implement the main strategy involving the killing points.

Let s be any stage. Recall that each type-M node has a sequence of partial trace functions. At stage $s + 1$, we first examine all such traces which are currently defined in increasing order. Whenever the computations associated with a trace change, the trace should enter C. Since the trace may be prevented from immediately doing so, we must mark the fact that it desires to enter C, and continue monitoring the situation as long as it remains out, hoping to enumerate it if at all possible. Suppose $c = c^\beta(k)[s]$, and let $t \leq s$ be the stage at which $c^\beta(k)[s]$ was chosen. Suppose that either A or B has changed on the use for k: that is, either

$$A[s] \upharpoonright \phi^\beta(A \oplus C; k)[t] \neq A[t] \upharpoonright \phi^\beta(A \oplus C; k)[t], \text{ or}$$

$$B[s] \upharpoonright \phi^\beta(B \oplus C; k)[t] \neq B[t] \upharpoonright \phi^\beta(B \oplus C; k)[t].$$

For all $s' > s$ we say c *wishes to enter* C *at* s'. Additionally, for all $s' > s$, all α and l such that $\beta^\frown\infty \subseteq \alpha$ and $c^\beta(k) < c^\alpha(l)[s]$, $c^\alpha(l)[s]$ *wishes to enter* C *at* s'. This is because in this case we must also try to enumerate $c^\alpha(l)$ into C in order to ensure $\phi^\beta(k) < c^\alpha(l)$ if we can. There is no need to worry about $c^\alpha(l)$ for the sake of the β-strategy itself, since $c^\beta(k)$ already wishes to enter; however, negative strategies which take $\phi^\beta(k)$ into account still need to be protected from having to take $\phi^\alpha(l)$ into account as well if at all possible. This action causes injury to α's strategy for which β must take responsibility. Below in 3.5 we describe a trace-replacement procedure that forces β to correct such a situation.

We must enumerate such a c into C at stage $s+1$ if possible. However, even if c cannot enter C, we still must initialize all nodes η which were counting on the stability of $\phi(k)$. Hence we initialize all $\eta \in \mathcal{T}[s]$ such that the initialization of $\beta^\frown\langle k+1, n\rangle$ for every n initializes η at $s+1$.

For each c that wishes to enter C at stage $s+1$, we examine all currently defined restraints that have previously been set by some α with authority to restrain c at s. Recall that such an α has restraint functions $r^\alpha(y)[s]$ for each $y \in \omega$, and a total restraint function, $r^\alpha[s] = \max(\{\, r^\alpha(y)[s] \,:\, y \in \omega \,\})$. For each y such that $r^\alpha(y)\downarrow[s]$, let $s^\alpha(y)[s]$ be the least t such that for all t' with $t < t' \leq s$, $r^\alpha(y)\downarrow[t'] = r^\alpha(y)[s]$. In other words, $r^\alpha(y)[s]$ was last set at $s^\alpha(y)[s]+1$. A subsequent change in the computation on y involved in α's requirement below this restraint should remove it and thereby allow traces to enter C. If either

1. α has requirement N^A_Ψ and
$$A \restriction \psi^+(A \oplus C; y)[s^\alpha(y)[s]] \neq A \restriction \psi^+(A \oplus C; y)[s],$$
or
2. α has requirement N^B_Ψ and
$$B \restriction \psi^+(A \oplus C; y)[s^\alpha(y)[s]] \neq B \restriction \psi^+(B \oplus C; y)[s],$$

then $r^\alpha(y)\uparrow[s+1]$.

We must initialize any $\eta \in \mathcal{T}[s]$ which acted on the assumption that $r^\alpha(y)\downarrow$. Hence, we initialize all η such that the initialization of $\alpha^\frown y$ initializes η at $s+1$.

Suppose $c = c^\beta(k)[s]$. If c wishes to enter C at $s+1$ and
$$\max\{\, r^\alpha(y)[s+1] \,:\, \alpha \text{ has authority to restrain } c \text{ at } s+1 \,\} < c,$$
then $c \in C[s+1]$. We must also allow all traces to enter that are associated with computations with use high enough to be affected by the entry of c. If $d > c$ with $d = c^\alpha(l)[s]$ for some l and α and $(c^\beta(k) < \max\{\phi^\alpha(A \oplus C; l), \phi^\alpha(B \oplus C; l)\})[s]$, then $d \in C[s+1]$. For any d which enters C at $s+1$, we must initialize all $\eta \in \mathcal{T}[s]$ which were counting on $d \notin C$. Suppose $d = c^\alpha(l)[s]$. Then, for all $l' \geq l$, $c^\alpha(l')\uparrow[s+1]$, and we initialize all η such that $\alpha^\frown\langle l', l'\rangle$ initializes η at $s+1$.

If any such initialization or enumeration takes place then the stage is destructive and after all such enumerations and initializations have been finished, we end stage $s+1$ and go to stage $s+2$. Otherwise, the stage is constructive and we begin the recursive calculation of the current approximation to the true path g. Beginning with $g[s] \upharpoonright 0 = \lambda$, we let α be $g[s] \upharpoonright n$ and continue as in sections 3.4, 3.5, and 3.6 below.

3.4 Strategies for incomparability requirements

Suppose α has requirement N^A_Ψ assigned to it. The action for α if α has some requirement N^B_Ψ assigned to it will be exactly the same as that described below, with A and B interchanged everywhere. Let s^- be the last α-stage or 0 if there has been no previous α-stage. If $s^- = 0$, then we immediately end stage $s+1$ and go to stage $s+2$. Thus, all α does at the successor to the first α-stage is to take its place in the prior-to relation.

Otherwise, suppose s is not the first α-stage.

If there is some y such that $r^\alpha(y)[s] < r^\alpha(y)[s^-+1]$, some computation associated with α's y-computation has turned out to be wrong. Although this could have been caused by an honest change in $A \upharpoonright \psi(A \oplus C; y)$, it could also have been the result of a change in a computation involving some $\beta^\frown\infty \subseteq \alpha$. In this case, as explained at the end of section 2.1, the restraint should be reset for any such y. For each y such that $r^\alpha(y)\!\downarrow\![s^-+1]$, $r^\alpha(y)\!\uparrow\![s]$, but $A \upharpoonright \psi(A \oplus C; y)[s^-] = A \upharpoonright \psi(A \oplus C; y)[s]$, we let $r^\alpha(y)[s+1] = \psi^+(A \oplus C; y)[s]$.

As mentioned in section 3.2, some nodes with ghosts on α may interfere with α's belief in a computation at $s+1$. We only consider Ψ to converge on computations which are not interfered with in this way. With this convention, we let

$$l^\alpha[s] = \max\{\, x \,:\, \forall y \le x(\Psi(A \oplus C; y)\!\downarrow = B(y))[s] \,\}.$$

We let $l^\alpha[s] = -1$ if no x as above exists. We say s is α-expansionary if s is an α-stage and for all α-stages $t < s$, $l^\alpha[t] < l^\alpha[s]$. Let m be least such that $r^\alpha(m)\!\uparrow\![s]$. If $r^\alpha(m)\!\uparrow\![s^-+1]$, we may wish to extend the α-strategy's threat to compute B from A. (If $r^\alpha(m)\!\downarrow\![s^-+1]$, then it appears that B cannot be computed from $A \oplus C$ using Ψ, so of course we take no further action.) If s is α-expansionary, then let $r^\alpha(m)[s+1] = \psi^+(A \oplus C; m)[s]$ (Notice that $m \le l^\alpha[s]$.) Let $r^\alpha[s+1] = \max(\{\, r^\alpha(y)[s+1] \,:\, y \le l^\alpha[s]\})$.

If n is least such that $r^\alpha(n)\!\uparrow\![s+1]$ we let $\alpha^\frown n$ be accessible at stage $s+1$. Notice that if $r^\alpha(n-1)$ has been reset at $s+1$ when it was undefined at s

merely because of the instability of some computation involving the trace of some β with $\beta^\frown\infty \subseteq \alpha$, then $\alpha^\frown n$ will have been initialized since it last was accessible. We show below that if α is on the true path this can only happen a finite number of times. It is necessary to give α outcome n in this case to prevent nodes extending $\alpha^\frown n-1$ from restraining additional traces which may lie below $\psi^+(n-1)$, which could cause $\psi^+(n-1)$ to increase by the recursive clause of Definition 3.4. This would introduce infinite injury from below, and so must be avoided. This same feature shows up in a somewhat more complicated way in the outcomes of ghosts of α, as explained in section 3.6 below.

3.5 Strategies for infimum requirements

Suppose α has a strategy for requirement M_Φ assigned to it. As in the case of the type-N nodes in 3.4 we only consider Φ to converge on computations when it does so without interference from nodes with which it has infinitary conflict. With this convention, we let

$$l^\alpha[s] = \max\big\{\, x \,:\, \forall y \leq x(\Phi(A \oplus C; y)\!\downarrow = \Phi(B \oplus C; y)\!\downarrow)[s] \,\big\}.$$

We let $l^\alpha[s] = -1$ if no x as above exists, and define a stage s to be α-expansionary as above: s is an α-stage and for all α-stages $t < s$, $l^\alpha[t] < l^\alpha[s]$. We define a partial trace-function c^α below.

Let s^- be the last α-stage before s, or 0 if no such stage exists. If α has not acted by stage s, so that $s^- = 0$, we immediately end stage $s+1$ and proceed to stage $s+2$. Thus, as in section 3.4, the only effect on α of this stage is the taking of its place in the prior-to relation.

Otherwise, suppose $s^- > 0$.

If any α-trace c is held out of C because of some restraint $r^\beta(n)$, yet the ghost of β included in α has an outcome $m \leq n$, then the current state of the construction disagrees with α's guess about β. This is similar to the interference with restraint-setting caused by the outcomes of ghosts: if α's guess is true, eventually β will give up its hold on the trace, and allow α to redefine the associated computation correctly. So α must wait for c to enter before taking any further action. More precisely, suppose there exist a type-N node β, a ghost, β_0, of β and numbers l and n such that

1. β has authority to restrain $c^\alpha(l)[s]$ at s,

2. $c^{\alpha}(l)[s]$ wishes to enter C at s,

3. n is least such that $(r^{\beta}(n) > c^{\alpha}(l))[s]$,

4. $\beta_0^\frown m \subseteq \alpha$, and

5. $m \leq n$.

We say β_0 *delays α's action at* $s+1$. If this is the case, we immediately end stage $s+1$ and proceed to stage $s+2$.

Otherwise, there are three possibilities. Recall from section 2.2, that infimum strategies that fail to eventually define a correct $A \oplus B$-computable functional have a double outcome: the least number k on which their functional fails and the possibly greater number l of the smallest trace that is kept out of C permanently. Let l be least so that $c(l)$ is undefined at stage s.

If there exists some least $k \leq l$ such that either

$$(A \oplus C)[s] \restriction \phi(A \oplus C; k)[s^-] \neq (A \oplus C)[s^-] \restriction \phi(A \oplus C; k)[s^-], \text{ or}$$

$$(B \oplus C)[s] \restriction \phi(B \oplus C; k)[s^-] \neq (B \oplus C)[s^-] \restriction \phi(B \oplus C; k)[s^-],$$

then we take no action for the α-strategy and let $\alpha^\frown\langle k, l\rangle$ be accessible at stage $s+1$.

Otherwise, if s is not α-expansionary, we let $\alpha^\frown\langle l, l\rangle$ be accessible at stage $s+1$.

Finally, suppose s is α-expansionary. In this case, we would like to choose a new trace for the α-strategy, and then allow $\alpha^\frown\infty$ an opportunity to act. Before we let $\alpha^\frown\infty$ be accessible, we may also need to replace some traces belonging to lower-priority nodes which were enumerated solely to limit the injury inflicted on other yet lower priority type-N strategies by an unstable computation associated with α. Our definition is complicated because we only wish to hold a node responsible for choosing new traces that no lower-priority node might have also affected since the last $\alpha^\frown\infty$ stage.

Let $s_0 \leq s$ be the stage at which $\alpha^\frown\infty$ was last accessible, and suppose there exist γ extending $\alpha^\frown\infty$, k_0, l_0, and $t < s_0$ such that

(1.) $c^{\gamma}(k_0)\uparrow[s]$,

(2.) $c^{\gamma}(k_0)\uparrow[t]$,

(3.) $c^\gamma(k_0){\downarrow}[t+1] = c^\gamma(k_0)[s_0]$,

(4.) $(A \oplus C)[s] \restriction \phi^\gamma(A \oplus C; k_0)[t] = (A \oplus C)[t] \restriction \phi^\gamma(A \oplus C; k_0)[t]$,

(5.) $(B \oplus C)[s] \restriction \phi^\gamma(B \oplus C; k_0)[t] = (B \oplus C)[t] \restriction \phi^\gamma(B \oplus C; k_0)[t]$,

(6.) γ has not been initialized at any stage t' with $t \leq t' \leq s$.

(7.) $(c^\alpha(l_0) < c^\gamma(k_0))[t+1]$,

(8.) either

  (a.) $(A \oplus C)[s] \restriction \phi(A \oplus C; l_0)[t] \neq (A \oplus C)[t] \restriction \phi(A \oplus C; l_0)[t]$, or
  (b.) $(B \oplus C)[s] \restriction \phi(B \oplus C; l_0)[t] \neq (B \oplus C)[t] \restriction \phi(B \oplus C; l_0)[t]$, and

(9.) for all $\beta \in \mathcal{T}[s]$ and $m_0 \in \omega$ such that $\alpha^\frown\infty \subseteq \beta^\frown\infty \subseteq \gamma$ and $(c^\beta(m_0) < c^\gamma(k_0))[t+1]$, both

  (a.) $(A \oplus C)[s] \restriction \phi^\beta(A \oplus C; m_0)[t] = (A \oplus C)[t] \restriction \phi^\beta(A \oplus C; m_0)[t]$, and
  (b.) $(B \oplus C)[s] \restriction \phi^\beta(B \oplus C; m_0)[t] = (B \oplus C)[t] \restriction \phi^\beta(B \oplus C; m_0)[t]$.

Then α *has responsibility for* $c^\gamma(k_0)[t+1]$ *at* s *because of* $\phi(l_0)$.

The last two clauses ensure that the correct node gets the responsibility of replacing the lost trace. The choice of $t+1$ as the first stage at which the lost trace converged protects α from infinite injury from below.

If there do not exist any traces for which α has responsibility at s, then let $c^\alpha(l)$ be the least number greater than any yet mentioned in the construction and let $\alpha^\frown\infty$ be accessible at stage $s+1$.

Otherwise, there must exist some least $l_0 \leq l$ such that α has responsibility for some trace at s because of $\phi(l_0)$. If $l_0 = l$, then we wish to preserve the relationship between $c^\alpha(l)$ and these traces so that the continual instability of $\phi^\alpha(l)$ will prevent the strategies associated with such traces from inflicting infinite injury on lower-priority strategies to the left of $\alpha^\frown\langle l{+}1, 0\rangle$. In this case, we first let $c^\alpha(l)$ be the least number greater than any yet mentioned in the construction before any trace replacement for lower-priority nodes takes place. In any event, we next replace all traces for which α has responsibility (for any $\phi(l_0)$), in order of their values at the last $\alpha^\frown\infty$-stage, choosing each in turn greater than any yet mentioned in the construction. Finally, if $c^\alpha(l)$ was not defined previously to this procedure, then we let $c^\alpha(l)$ be the least number greater than any yet mentioned in the construction. We let $\alpha^\frown\infty$ be accessible at stage $s+1$ and continue as in section 3.6 below.

3.6 Ghost outcomes

We call the outcomes of sections 3.4 and 3.5 *actual outcomes*, since in each case they represent the approximation to the true final state of the α-strategy. With the exception of the outcome ∞, after each actual outcome of a type-M strategy α must come a sequence of ghost outcomes. Suppose the outcome of α at $s+1$ is $\langle k_0, l_0\rangle$. Let $\beta_0, \dots, \beta_m$ be the nodes extending $\alpha^\frown\infty$ which have killing point less than k_0, in order of increasing killing point. We define a finite ghost-sequence δ^α of length $m+1$ recursively. For $i \leq m$, let s_i^- be the last previous $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$-stage.

Suppose β_i has requirement N^A_Ψ assigned to it. (The procedure for a type-N^B requirement has an analogous definition with the obvious changes.) If β_i has been initialized since this point was last accessible, then the outcome at $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$ is 0. Otherwise, let n be least such that there exists t with $s^- < t \leq s$ and $r^{\beta_i}(n)\big\uparrow[t]$. Since β_i has not been initialized since this point was last accessible, there are three possible reasons for this: either the β_i-strategy appears to be succeeding on n, or only infimum requirements prior to both β_i and α have injured the β_i-strategy on n, or the β_i-strategy has been injured by α itself or some infimum strategy extending $\alpha^\frown\infty$. In the second case, the injury appears to be merely temporary, since eventually all traces affecting the nth attempt at preservation by β_i will stabilize on their computations. In this case, the divergence of $r^{\beta_i}(n)$ appears to be false given the other information available at $s+1$ from the outcomes included in α. Because of this, the outcome at $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$ should be $n+1$ rather than n. In the third case, the noninfinitary outcome of α itself indicates that $r^{\beta_i}(n)$ is destined to be undefined infinitely often, so that the outcome should be n, just as it is in the case where the nth attempt by the β_i-strategy is actually successful. More precisely, if $A[s] \restriction \psi(A \oplus C; n)[s^-] \neq A[s^-] \restriction \psi(A \oplus C; n)[s^-]$, or if there exists some β with $\beta^\frown\infty \subseteq \beta_i$ and $\beta^\frown\infty \not\subseteq \alpha$ and some $k \in \omega$ such that $(\phi^\beta(k) < r^{\beta_i}(n))[s]$ and either

$$A[s] \restriction \phi^\beta(A \oplus C; k)[s_i^-] \neq A[s_i^-] \restriction \phi^\beta(A \oplus C; k)[s_i^-], \text{ or}$$

$$B[s] \restriction \phi^\beta(B \oplus C; k)[s_i^-] \neq B[s_i^-] \restriction \phi^\beta(B \oplus C; k)[s_i^-],$$

then let $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$ have outcome n; otherwise, let $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$ have outcome $n+1$.

Suppose β_i has requirement M_Ξ. If there exists $k \in \omega$ and t with $s_i^- < t \leq s$ and either

$$(A \oplus C)[s] \restriction \xi(A \oplus C; k)[s_i^-] \neq (A \oplus C)[s_i^-] \restriction \xi(A \oplus C; k)[s_i], \text{ or}$$

$$(B \oplus C)[s] \restriction \xi(B \oplus C; k)[s_i^-] \neq (B \oplus C)[s_i^-] \restriction \xi(B \oplus C; k)[s_i],$$

then let k be least such. Otherwise, let k be least such that there exist t with $s^- < t \leq s$ and $c^{\beta_i}(k)\Big\uparrow$. In either case, let l be least such that there exists t with $s^- < t \leq s$ and $c^{\beta_i}(l)\Big\uparrow[t]$. Let $\alpha^\frown\langle k_0, l_0\rangle^\frown(\delta^\alpha \restriction i)$ have outcome $\langle k, l\rangle$

If $i = m$, we finally reach a nonghost node by this procedure. Let $\alpha^\frown\langle k_0, l_0\rangle^\frown\delta^\alpha$ be accessible at stage $s+1$.

This completes the construction.

4 Verification

We now prove that our construction results in a set C which satisfies all the requirements. We define a limit tree on which a true path, that is, a leftmost path of nodes which are accessible infinitely often, exists. Because ghost nodes representing nodes to the right of the true path lie on the true path, we have a little more work to do than is the case in other priority constructions in order to show that the true path actually exists, and that the nodes on it succeed in satisfying their requirement. This is because we must also show that those strategies which have a ghost on the true path, yet lie to the right of the true path, will not inflict infinite injury on strategies below their ghosts on the true path by the encoding of false information. Additionally, for this same reason, it is not immediately obvious that the lim inf of the ghost outcomes even lies on the tree, because it seems possible that this lim inf could be unbounded even if the requirement is satisfied by some other strategy. Thus it is not clear that the true path is infinite. We establish some of the easier technical facts about the true path first, then prove the lemmas characterizing the enumeration of traces in the construction, after which we go on to show that the requirements are satisfied. Since the situation for a type-N^B strategy is exactly the same as that for a type-N^A strategy, we merely prove that for each computable functional Ψ, $\Psi(A \oplus C) \neq B$, and that a ghost of a type-N^A node has an outcome on the true path. We then show that each infimum requirement is satisfied by the strategy of the associated node on the true path, and that ghost versions of type-M nodes on the true path have outcomes there. Finally, we show that $C \leq_T A \oplus B$.

We define the *limit tree*

$$\mathcal{T} = \left\{ \alpha : \exists s \, \forall t \geq s \, \alpha \in \mathcal{T}[t] \right\}.$$

We define the *true path*, g, by recursion using the equation $g(n) = \mathcal{O}$, where $\mathcal{O}$ is the leftmost outcome of $g \upharpoonright n$ accessible infinitely often in the construction, if such an outcome exists. We establish some basic properties of g with three lemmas proved by straightforward inductive arguments.

Lemma 4.1. $\forall n \, g \upharpoonright n \in \mathcal{T}$

Proof. Clearly $\lambda \in \mathcal{T}$. Suppose $g \upharpoonright n \in \mathcal{T}$, we have reached some stage s such that $g \upharpoonright n \in \mathcal{T}[t]$ for all $t \geq s$, and $g(n) = \mathcal{O}$. If $g(n)$ is a ghost outcome, then $g \upharpoonright n$ is a ghost, and hence none of its immediate outcomes are ever removed, since new ghost outcomes are only introduced to replace actual outcomes in section 3.1. So in this case, $g \upharpoonright n + 1 \in \mathcal{T}$. On the other hand, if $g(n)$ is an actual outcome, then once $g(n)$ is the outcome at some stage $t \geq s$, it cannot be removed from any $\mathcal{T}[t']$, for $t' > t$, since such a removal would result in a permanent change of $g \upharpoonright n$ to a ghost. But then the nonghost outcome $\mathcal{O}$ could never again be the outcome of $g \upharpoonright n$, contradicting the fact that $g(n) = \mathcal{O}$. Thus $\forall n (g \upharpoonright n \in \mathcal{T})$. □

We must check that every requirement gets assigned to a node on the true path. If g is infinite, this is guaranteed to happen. This follows because, on g, we are guaranteed that we have no infinite sequences of ghosts.

Lemma 4.2. *If g is infinite, then there exist infinitely many $\alpha \subset g$ which are actual nodes.*

Proof. Suppose $g \upharpoonright n$ is an actual node, with actual outcome $g(n)$. If $g \upharpoonright n$ is a type-N node, then $g \upharpoonright n + 1$ is an actual node, and the same holds if $g \upharpoonright n$ is type-M and $g(n) = \infty$. Suppose $g(n) = \langle k, l \rangle$. Note fewer than k nodes act before stage k. This means that fewer than k nodes can have killing point less than k on $g \upharpoonright n$, hence $g \upharpoonright n + 1$ must have fewer than k ghost outcomes in any sequence extending it before an actual outcome is introduced. □

We also need to show that once a node has been accessible, it can no longer cease to exist by somehow dropping from subsequent priority trees in our sequence. Otherwise our priority arrangement would be unstable.

Lemma 4.3. *If $\beta \subseteq g[s]$, then for all $t \geq s$, $\beta \in \mathcal{T}[t]$.*

Proof. Otherwise, suppose there exists a least $t > s$, such that $\beta \notin \mathcal{T}[t]$. Then some β_0 having infinitary conflict with β must have acted for the first time at t, and a ghost version of β_0 must have been introduced changing the tree $\mathcal{T}[t-1]$ at $\gamma = \beta \cap \beta_0$. Then, for some $k, l \in \omega$, $\gamma^\frown\langle k, l\rangle \subseteq \beta$ and $\gamma^\frown\infty \subseteq \beta_0$. Since $t - 1 \geq s > k$, the killing point of β_0 on γ is greater than k, so that no ghost of β_0 can be introduced extending $\gamma^\frown\langle k, l\rangle$. This is a contradiction. □

4.1 Trace movement

We must prove some lemmas which describe how restraints change to allow traces to enter C. It is useful to keep track of the weakest restraint on a trace at any given stage. We say β *restrains c at a stage s* if β has authority to restrain c at s and $r^\beta[s] \geq c$. If β restrains a trace c that wishes to enter C at stage s and every other node that restrains c is prior to β, we say β *holds c at s*. (Notice that restraining at s is a weaker notion than holding c at s.) First, we show that when a node holding a trace drops restraint, a prior node must hold the trace if it does not enter C.

Lemma 4.4. *Suppose c is any trace held by a node β at some stage s, and z is least such that $r^\beta(z)[s] \geq c$. If there exists a least stage $t > s$ such that $r^\beta(z)\uparrow[t]$, then either*

1. *some node β_0 prior to β holds c at t, or*
2. *$c \in C[t+1]$.*

Proof. Suppose that $c \notin C[t+1]$ and no prior node holds c at t. Then some subsequent β_0 must hold c at t. We rule out all possibilities for β_0. Note that β_0 cannot restrain c at s, since otherwise β_0 rather than β holds c at s. Let $s_0 \leq s$ be the stage at which $r^\beta(z)$ first converged. Notice that c is chosen before s_0. If β were to initialize β_0 at any stage t' with $s_0 < t' \leq t$, then β_0 would have no authority to restrain c at t by Definition 3.2. Since $r^\beta(z)$ is undefined at t, this implies that if $\beta <_L \beta_0$, β_0 must extend the infinite outcome of $\beta \cap \beta_0$. Since β is prior to β_0, β must have been first accessible before β_0 in the construction, and hence must extend an outcome of $\beta \cap \beta_0$ which has a first element smaller than β_0's killing point on $\beta \cap \beta_0$.

But then β_0 is initialized at the β-stage s_0, losing any authority to restrain c. So $\beta \not<_L \beta_0$. Clearly, $\beta_0 \not\subset \beta$, since β was accessible at an earlier stage. Suppose $\beta \subset \beta_0$. Recall that β_0 does not restrain c at s. Hence, if β_0 sets a restraint at some stage between s and t, it must extend some outcome of β to the right of that which β has at t. But then β_0 is initialized at t, and hence cannot restrain c at this stage. This leaves only the possibility that $\beta_0 <_L \beta$. Let t_0 be the β_0-stage at which β_0 sets its restraint holding c. First, suppose β is never initialized at any t' with $s < t' \leq t$. In this case, β is not initialized at the β_0-stage $t_0 \leq t$. Hence, since $\beta_0 <_L \beta$, β_0 must extend a ghost of β. If this ghost has outcome $l > z$, then β_0 is initialized at t and so cannot hold c. Suppose, on the other hand, that this ghost has outcome $l \leq z$. If β_0's ability to set a restraint is not interfered with by β according to Definition 3.3 (1), then the node β' that restrains c at t_0 and is prior to all others that do so must be different from β. As argued above in the case of β and β_0, either β' or some ghost of β' must be included in β_0, since otherwise the activity of one would initialize the other, so that the same trace could never be restrained by both of them. But then, since β' does restrain c at t_0, but does not interfere with β_0's ability to set a restraint, β_0 must have been initialized at some stage before the smallest β' restraint bigger than c was set, and been inaccessible until some stage after the one at which that restraint was set. But this is a stage after that at which c was chosen. This contradicts the assumed authority of β_0 to restrain c in the first place. Finally, suppose β is initialized at some t' with $s < t' \leq t$. Then $t = t'$, since $r^{\beta}(z)\uparrow[t']$. But no restraint can be set at a stage when nodes are initialized, since this can only happen during a destructive stage, and, if this happens, then no further restraint setting action is taken until the next stage. $\square$

Now we can show that gradually, nodes that first acted earlier and earlier in the construction must be the ones setting the only restraints that matter.

Lemma 4.5. *Suppose β holds c at stage s and $t \geq s$ with $c \notin C[t+1]$. Then there exists some β' equal to or prior to β such that*

1. *β' restrains c at t, and*
2. *there exists some t' with $s \leq t' \leq t$ such that β' holds c at t'.*

Proof. Suppose not, and choose t least so that this fails. Since β itself satisfies the claim at s, $t > s$. Hence there is some β' equal to or prior to β such that

β' restrains c at $t-1$ and there is some $t' \leq t-1$ such that β' holds c at t'. Hence β' must fail to restrain c at t. But then by Lemma 4.4, some β_0 prior to than β' must hold c at t. Since such a β_0 must be prior to β as well, this is a contradiction. $\square$

We next show that the restraint setting procedure for a node β that is not initialized infinitely often is eventually adequate.

Lemma 4.6. *Let β have type N and let $s_0 < t$ be such that β is never initialized at any stage s' with $s_0 \leq s' \leq t$. Let $t > s > s_0$ and suppose k is such that for all t' with $s \leq t' \leq t$, $r^\beta(k)\downarrow[t']$, $r^\beta(k)\uparrow[s-1]$, no trace appointed before s_0 enters C after stage s, and either*

1. *β has type N^A, and $A[t] \restriction r^\beta(k)[s] = A[s] \restriction r^\beta(k)[s]$; or*
2. *β has type N^B, and $B[t] \restriction r^\beta(k)[s] = B[s] \restriction r^\beta(k)[s]$.*

Then $C[t] \restriction r^\beta(k)[s] = C[s] \restriction r^\beta(k)[s]$.

Proof. Otherwise there exists some least $c < r^\beta(k)[s]$ that entered C between s and t. c must be a trace that was appointed at some stage $s' < s$, since all traces appointed after stage s are greater than $r^\beta(k)[s]$. Also $s_0 \leq s'$, by the assumptions on s. Clearly, c is still current at stage s, so suppose $c = c^\gamma(n)[s]$. If β has authority to restrain c at s, then c cannot have entered $C[t]$, since β's restraint never drops back between s and t. Hence β must fail to have authority to restrain c at s. We show that this is impossible by examining all the possible relationships between γ and β. If $\beta <_L \gamma$, then γ must be prior to β, since γ is not initialized between s' and s. This follows since all nodes to the right of β except those which are prior to β are initialized at some stage prior to s (by some outcome of either β or a node which it extends) and remain inaccessible until some stage after $r^\beta(k)$ is defined at s. In this case β must include a ghost of γ. But then either γ interferes with β's ability to set a restraint at s via this ghost, or β is initialized when c first wishes to enter C, or β is initialized when c actually enters C. All three are contrary to hypothesis. If $\gamma <_L \beta$, then, by our assumption on s_0, γ must include a ghost of β — otherwise β is initialized when c enters C by Definition 3.5, either (1) or (4). Hence either $\gamma \subset \beta$, or $\beta \subset \gamma$, or $\gamma <_L \beta$ with a ghost of β on γ. First, suppose $\gamma^\frown\langle k', l\rangle \subseteq \beta$ for some $k', l \in \omega$. In this case, by section 3.3, $k' \leq l \leq n$, since β is never initialized between s_0 and t. But then $c^\gamma(n)\uparrow[s]$, which is a contradiction. So, either $\gamma^\frown\infty \subseteq \beta$, or $\beta \subset \gamma$, or $\gamma <_L \beta$, with a ghost $\beta_g \subset \gamma$. But then, since c was appointed after β was last initialized, by the assumptions on s_0, β would have authority over c after all. $\square$

We now show that any type-N^A strategy can only impose finite restraint on the construction. Naturally, this is only important if the node is accessible infinitely often after last being initialized. Of course, this also holds for type-N^B strategies.

Lemma 4.7. *Let β be a node with requirement N^A_Ψ assigned to it. Then $\liminf_{s\to\infty} r^\beta[s] < \infty$.*

Proof. Suppose not and so,in particular, β is initialized only finitely often. For every y, $r^\beta(y)$ can only increase at a stage $s+1$ if it is undefined at stage s. Hence, by hypothesis, every restraint $r^\beta(y)$ is eventually set permanently. Clearly this can only happen if $\Psi(A \oplus C) = B$. We now define a total functional which computes B from A. Let t_0 be the stage at which β is last initialized. For each k, let $s(k) > t_0$ be the stage at which β permanently sets its restraint $r^\beta(k)$. Without loss of generality, we can assume for simplicity's sake, that for all k, $C \restriction t_0 = C[s(k)] \restriction t_0$, even though there may be some finitely many k for which this fails. By Lemma 4.6, then, for all $k \in \omega$, $\Psi(A \oplus C; k) = \Psi(A \oplus C)[s(k)]$. Since the requirement fails, this means $B(k) = B(k)[s(k)]$. But then, since $\lambda k.s(k) \leq_T A$, $B \leq_T A$. This is a contradiction. $\square$

4.2 Satisfaction of requirements

We still must show that the true path g is infinite. We also must show that every substring of g is initialized only finitely often, imposes only finitely much restraint below itself, and succeeds in satisfying the associated requirement. As indicated above, not only does this mean we must show that the negative requirements are satisfied, but we also must show that for every type-M node γ with a ghost on the true path there is some number m such that for infinitely many stages s $c^\gamma(m)\uparrow[s]$. This shows that the sequence of ghost outcomes for γ will come to a limit. We therefore assume this as an inductive hypothesis in what follows.

Suppose $\alpha = g \restriction n$. Below, we assume that every stage mentioned is greater than the last stage at which α is initialized, and is great enough so that α lies on the priority tree at that stage. Notice that any type-M $\beta \subset \alpha$

with $\beta^\frown\infty \subset \alpha$ must eventually stabilize on all its traces by definition of the assignment of outcomes.

4.2.1 Satisfaction of N^A_Ψ

Suppose α has requirement N^A_Ψ assigned to it or is a ghost of some node with requirement N^A_Ψ. By Lemma 4.7, no type-N node can impose infinite restraint. If α is a ghost, there is therefore nothing to show here, since the sequence of α's outcomes has a finite liminf.

This leaves the case where α is an actual node. We suppose for the sake of a contradiction that $\Psi(A \oplus C) = B$. Although α is on the true path, it is not true that nodes to the left of α act only finitely often to appoint traces or set restraints. What is true, however, is that there are only finitely many stages at which nodes prior to and to the left of α act. If t_0 is the α-stage just after the stage at which α is last initialized, then we may assume without loss of generality that no such node is accessible after t_0.

Let $k \in \omega$, and suppose for all $m \leq k$, $(\Psi(A \oplus C; m)\downarrow= B(m))[s]$. Since the requirement is not satisfied, this must hold at cofinitely many stages s. Showing that $r^\alpha(k)$ is defined in the limit will be sufficient for a contradiction, since then it will follow by a straightforward induction that $\lambda y.r^\alpha(y)$ is total, and hence $\lim\limits_{s\to\infty} r^\alpha[s] = \infty$, contrary to Lemma 4.7.

In order to make what follows more readable, throughout this section, we abbreviate the functionals involved by omitting the oracle $A \oplus C$; in other words we just write $\psi(k)$ instead of $\psi(A \oplus C; k)$. We also do this for the increased use ψ^+. In the case of α, this should cause no confusion, since $A \oplus C$ is the only oracle which we care about for these functionals, but we also do this for type-M nodes. Throughout this section we are almost always only concerned with the effect of the changes in the oracle $A \oplus C$, but we make both of the oracles $A \oplus C$ and $B \oplus C$ explicit whenever there is any danger of confusion.

Let s_0 be an α-stage such that for all $s \geq s_0$, $(\forall m < k)\, r^\alpha(m)\downarrow[s]$ and $(A \upharpoonright \psi(k))[s_0] = A[s] \upharpoonright \psi(k)[s_0]$. By the assignment of outcomes in section 3.5, any type-M node β such that $\beta^\frown\infty \subseteq \alpha$ must eventually stabilize on all $\phi^\beta(l)$. Because α is on the true path, to show that $\psi^+(k)$ eventually gets set permanently to the same value it is therefore sufficient to show that for every such β there are only a finite number of traces $c^\beta(n) < \psi^+(k)$. We can show that the procedure by which instability at higher-priority nodes causes the enumeration and replacement of lower-priority traces ensures that gradually nodes with higher and higher priority are responsible for all the instability in $r^\alpha(k)$, so that eventually no more traces appear below $\psi^+(k)$. Suppose this fails, and let β be the longest node with infinitely many traces below $\psi^+(k)$ such that $\beta^\frown\infty \subseteq \alpha$. More precisely, suppose $s_1 \geq s_0$ and for all γ with $\beta^\frown\infty \subseteq \gamma$ and $\gamma^\frown\infty \subseteq \alpha$ there exists some l^γ such that for all α-stages $s \geq s_1$, and all $l \in \omega$, if $c^\gamma(l)\!\downarrow\![s]$, then

1. if $l > l^\gamma$, $(c^\gamma(l) > r^\alpha(k))[s]$, and
2. if $l \leq l^\gamma$, $A[s] \upharpoonright \phi^\gamma(l)[s_1] = (A \upharpoonright \phi^\gamma(l))[s_1]$.

We need to show that these conditions hold also for β, with an appropriate choice of l^β and stabilization stage s^β in place of s_1. This will contradict the choice of β.

First note that if $y \leq k$, then no node η such that $\alpha^\frown y \subseteq \eta$ can act after stage s_0. By the assignment of ghost outcomes, for all $y \leq k$, and any ghost α_g of α, a node η that extends both $\beta^\frown\infty$ and $\alpha_g{}^\frown y$ can be accessible at most once after stage s_1, so we can assume without loss of generality that in fact no such node is ever accessible after stage s_1.

If $r^\alpha(k)\!\downarrow\![s]$ for all $s \geq s_1$, then both (1.) and (2.) above hold also for β, choosing l^β greatest such that $r^\alpha(k) \geq c^\beta(l^\beta)$. So suppose that there exists some α-stage $s_2 \geq s_1$ such that $r^\alpha(k)\!\uparrow\![s_2]$. Notice that $r^\alpha(k)\!\downarrow\![s_2+1]$ by section 3.4. Let l^β be greatest such that either

1. $c^\beta(l^\beta)[s_2+1]$ is held by some node at s_2+1,
2. $(c^\beta(l^\beta) \leq r^\alpha(k))[s_2+1]$, or
3. α has no authority to restrain $c^\beta(l^\beta)$ at s_2+1.

We show that for all $s > s_2$, for all $l > l^\beta$, if $c^\beta(l)\!\downarrow\![s]$ and $r^\alpha(k)\!\downarrow\![s]$, then $(r^\alpha(k) < c^\beta(l))[s]$. Once we show this, it is straightforward to show that there exists s^β such that the above conditions (1.) and (2.) hold for β in place of γ since β is on the true path, and hence (2.) will eventually hold.

Let t be the next α-stage after s_2 at which the above fails. Let $l' > l^\beta$ be least such that $(c^\beta(l') \leq r^\alpha(k))[t]$. Since $(c^\beta(l') > \psi(k))[s_0]$, there must, by choice of s_1 exist $\gamma \subseteq \beta$ and m such that $\gamma^\frown\infty \subseteq \alpha$ and $(c^\gamma(m) < c^\beta(l') < \phi^\gamma(m) < \psi^+(k))[t+1]$. This can only happen if $\phi^\gamma(m)$ increases at some stage $t^- \leq t$ after $c^\beta(l')$ is chosen, and α already must respect $\phi^\gamma(m)$ for

$\psi(k)$ at t^-. Then, at t^-, $c^\beta(l')[t+1]$ will wish to enter C, and $r^\alpha(k)\!\uparrow\![t^-]$. By assumption, $c^\beta(l')[t+1]$ does not enter C before $t+1$, so some η prior to α must restrain $c^\beta(l')$ at t. We show that such an η cannot exist, which establishes the contradiction.

If $\eta <_L \alpha$ and there are no infinitary conflicts between them or if $\eta \subset \alpha$, then α would be initialized at the stage at which η's restraint increases above $c^\beta(l')$, and hence have no authority over $c^\beta(l')$. Since any stage at which α was initialized would have to be before s_0, $c^\beta(l')\!\downarrow\![s_2+1]$, contradicting clause (c) of the choice of l^β. If $\alpha <_L \eta$ and there are no infinitary conflicts between them, if $\alpha \subset \eta$, or if some ghost of α is included in η, then η would be initialized by the initialization of $\alpha^\frown k{+}1$ at the stage t^-. This follows because, by (b) above, $c^\beta(l')$ is not held at s_2+1, hence $t > s_2 > s_0$, so η cannot have acted if $\alpha^\frown k' \subseteq \eta$ for some $k' \leq k$. Therefore, η can have no authority over $c^\beta(l')$.

Finally, suppose there is some infinitary conflict between α and η. If some ghost of η is included in α, then either α would be initialized at the stage at which η's restraint rises above $c^\beta(l')$, or interference from the outcome of this ghost would prevent α from setting a restraint at $t+1$. Both are contrary to the hypotheses. Hence η cannot exist, and $c^\beta(l')[t+1]$ cannot be below $r^\alpha(k)$. As discussed above, routine induction now establishes that $\psi^+(k)$ has a finite limit.

It is, however, still not obvious that there exists a sufficiently large s such that $r^\alpha(k)\!\downarrow\![s+1]$, since this restraint may have to wait to be set because of the interference of some β of higher priority, as defined in section 3.4 above. If such a β is a type-M node interfering by 3.3(2a), then the interfering c^β will have entered or will wish to enter by the next α-stage. If β interferes by 3.3(2b), then the troublesome $c^\beta[s]$ grows without bound, by the definition of ghost outcomes. Hence, this interference must eventually go away, since $\psi^+(A \oplus C; k)$ eventually gets set permanently to some finite value. It may not be so clear what happens in the case of a type-N β. Suppose some β interferes with α by 3.3, (1). Since such a β has a ghost along α, which is on the true path, the β-restraint involved will be undefined at some intervening stage before the next α-stage. After this point, there must always be some node prior to β which restrains c, by a straightforward induction using Lemma 4.5. So β can never interfere with $r^\alpha(k)$ because of this trace again. Hence, eventually $r^\alpha(k)\!\downarrow$, which is enough to show that N^α_Ψ is satisfied as discussed above.

4.2.2 Satisfaction of M_Φ

Suppose that α has requirement M_Φ assigned to it, or is a ghost of some actual node α_0 with requirement M_Φ. Because the order type of α's outcomes is either ω^2+1, or ω^2 (in the ghost case), it is not obvious that the sequence of α's outcomes have a $\liminf$ on $\mathcal{T}$, so that, as above, we have extra work to show that g does not stop with α. Of course, if α is an actual node, we also need to consider each possible outcome on g in order to verify that α's requirement is satisfied and that α does not keep lower-priority nodes on g from satisfying their requirements.

First, suppose α is a ghost of some α_0 which is not on the true path. If α_0 is not accessible infinitely often, then α has a finite outcome on the true path and there is nothing to show, so we assume otherwise. Let $\gamma = \alpha \cap \alpha_0$, and suppose $\gamma^\frown\langle k', l'\rangle \subseteq \alpha$. (Recall $\gamma^\frown\infty \subseteq \alpha_0$.) By Lemma 4.7, every type-N node restrains only finitely many traces permanently. Since α_0 is not on the true path, all but finitely many nodes extending $\alpha_0^\frown\infty$ are initialized at infinitely many stages s. In particular, almost all nodes with authority over α_0's traces are initialized whenever either $\phi^\gamma(A\oplus C; k')$ or $\phi^\gamma(B\oplus C; k')$ changes. By the trace-choosing and trace-replacement procedures, there must exist a stage t_0 such that for all $t \geq t_0$, if t is a $\gamma^\frown\infty$-stage, then $c^\gamma(k')$ is defined at $t+1$. Since the computations associated with any $k'' < k'$ are eventually stable, it follows from the trace-replacement procedure in section 3.5 that there exists some l_0 such that for all $l > l_0$, $c^\gamma(k') < c^{\alpha_0}(l)$ at every such stage $t+1$. But then, since $\gamma^\frown\langle k, l\rangle$ is on the true path, all such traces eventually wish to enter C. Only finitely many can be below the maximum of all the restraints of nodes with permanent authority over them, since there are only a finite number of these, as pointed out above. So there exists some least $l > l_0$ such that $c^\gamma(l)\!\uparrow$ at infinitely many stages s, and hence for some $k \leq l$, $\alpha^\frown\langle k, l\rangle$ is on the true path.

Next, suppose α is not a ghost. Suppose first that $\alpha^\frown\infty$ is not on the true path. Then there must exist some fixed k such that at infinitely many stages s there exists some l with $\alpha^\frown\langle k, l\rangle$ accessible at s. Since this cannot happen if $\Phi(A\oplus C; k) = \Phi(B\oplus C; k)$, α's requirement is satisfied, and we only need to show that its outcomes have a $\liminf$. As above, since there are only finitely many nodes extending $\alpha^\frown\infty$ with killing point less than k, all but finitely many traces c for α must enter C. Hence there is in fact some fixed l such that α gets outcome $\langle k, l\rangle$ at infinitely many stages s.

Finally, suppose $\alpha^\frown\infty$ is on the true path. In this case, $\Phi(A\oplus C) =$

$\Phi(B \oplus C)$. We show $\Phi(A \oplus C) \leq_T C$ using $\lim_{s\to\infty} c^\alpha(k)[s] + 1$ as the use of the reduction. We first show that this limit exists, then that $\Phi(A \oplus C; k)$ must have the same value that it does when $c^\alpha(k)$ takes on its final value. (Note that if this limit exists then it is computable from C as $c^\alpha(k)$ can change only after its current value enters C.)

The proof is by induction on k. Fix k and suppose that we have reached a stage s_0 such that $A \oplus C$ and $B \oplus C$ never change on the uses of $\Phi(A \oplus C; k)$ and $\Phi(B \oplus C; k)$, respectively, after stage s_0, and such that for all $l < k$, $c^\alpha(l)$ has reached its final value. Since $\phi(A \oplus C; k)$ and $\phi(B \oplus C; k)$ never change after s_0, no trace smaller than the maximum of these two uses can be enumerated into C at a subsequent stage. Hence, $c^\alpha(k)$ can only be enumerated because of some β with $\beta^\frown\infty \subseteq \alpha$. In other words, $c^\alpha(k)$ may wish to enter when an originally small use associated with a β-trace increases. We show $c^\alpha(k)$ must reach a final value by tracing backwards along the true path. Let $\beta^\frown\infty \subseteq \alpha$ and suppose we have reached a stage s^β such that for all γ and l, if $\beta^\frown\infty \subseteq \gamma$ and $\gamma^\frown\infty \subseteq \alpha$ and $(c^\gamma(l) < c^\alpha(k))[s^\beta]$, then $\phi^\gamma(A \oplus C; l)$ and $\phi^\gamma(B \oplus C; k)$ never again change after s^β. If this does not also hold for β in place of γ, then there must exist some l such that $(c^\beta(l) < c^\alpha(k))[s^\beta]$, $s > s^\beta$, and either

$$A[s] \upharpoonright \phi^\beta(A \oplus C; l)[s^\beta] \neq (A \upharpoonright \phi^\beta(A \oplus C; l))[s^\beta], \text{ or}$$

$$B[s] \upharpoonright \phi^\beta(B \oplus C; l)[s^\beta] \neq (B \upharpoonright \phi^\beta(B \oplus C; l))[s^\beta].$$

Choose l_0 least such and let $s_1^- > s^\beta$ be the least stage at which either of these uses changes, and s_1 be the next $\beta^\frown\infty$ stage. By the trace replacement procedure, there must exist some greatest $l_1 \geq l_0$ such that $c^\beta(l_1) < c^\alpha(k)$ at stage s_1. We claim that for all $l > l_1$ and all $\beta^\frown\infty$ stages $s \geq s_1^-$, $(c^\alpha(k) < c^\beta(l))[s]$. This holds at s_1 by trace replacement. By the hypotheses on β and s^β, $c^\alpha(k)$ can only increase because of a change in either $\phi^\beta(A \oplus C; l)$ or $\phi^\beta(B \oplus C; l)$ for some $l \leq l_1$, or in some computation associated with a trace for a strategy γ with $\gamma^\frown\infty \subseteq \beta$. In either case, any $c^\beta(l)$ for $l > l_1$ will be replaced after $c^\alpha(k)$ because the trace-replacement procedure preserves the original ordering relationship, or possibly allows $c^\alpha(k)$ to be replaced even earlier in the process, if ϕ^β changes below l_1 It now follows by a routine induction on β that for all β with $\beta^\frown\infty \subseteq \alpha$, there exist only finitely many traces for β less than $c^\alpha(k)$ in the limit. Hence $c^\alpha(k)$ is eventually defined permanently, since α is on the true path.

Let s be the least $\alpha^\frown\infty$ stage at which $c^\alpha(k)$ takes on its final value, so that $c^\alpha(k)[s] \notin C$. For convenience of notation, we write $c = c^\alpha(k)[s]$. We show that at any $\alpha^\frown\infty$ stage $t > s$, $\Phi(A\oplus C; k)[t] = \Phi(A\oplus C; k)[s]$. Suppose not, and let t_0 be the least such stage, and t^- be the greatest $\alpha^\frown\infty$ stage less than t_0. Then

$$\Phi(A \oplus C; k)[s] = \Phi(A \oplus C; k)[t^-] = \Phi(B \oplus C; k)[t^-], \text{ and}$$

$$\Phi(A \oplus C; k)[s] \neq \Phi(A \oplus C; k)[t_0] = \Phi(B \oplus C; k)[t_0].$$

This implies that both $A \oplus C$ and $B \oplus C$ have changed on their respective uses for Φ between t^- and t_0. Note that C cannot have changed on either of the uses $\phi(A \oplus C; k)[s]$ or $\phi(B \oplus C; k)[s]$, since any such change would cause c to enter C. Hence c must wish to enter C before t_0. Once c wishes to enter, any node preventing its entry successfully restrains numbers that could injure the relevant uses from entering C by Lemma 4.6, since the entry of any trace chosen before such a node was last initialized would cause the larger trace c to enter C as well if these uses were affected. Whether or not c first wishes to enter before t^-, this means that both A and B must have changed on the respective uses between t^- and t_0, causing any restraint set by a node extending $\alpha^\frown\infty$ which must respect c to become undefined. But then, since no node extending $\alpha^\frown\infty$ is accessible between t^- and t_0, none can hold c at t_0. By the delaying of action in Section 3.5, t_0 cannot be an $\alpha^\frown\infty$-stage if some higher priority node holds c. But then $c^\alpha(k)\uparrow[t_0]$, a contradiction.

4.3 The relative computability of C

We still must show that $C \leq_T A \oplus B$. We define a functional Γ such that $\Gamma(A\oplus B) = C$. Intuitively, this follows because once a trace c is chosen, it can only enter C because of changes in either A or B, and it can only be prevented from entering because of the failure of either A or B to change infinitely many times on some finite set of restraints. Proving this is only complicated by the fact that restraints can be canceled through the initialization of the nodes to which they are associated, so we must indicate how such initialization can also be detected by changes in some bounded part of $A \oplus B$.

We prove that we can compute the functional Γ by induction on c. Let $c \in \omega$, and suppose that we can compute $C \upharpoonright c$ using $\Gamma(A \oplus B)$. Then we can determine a stage s_0 such that for all $s \geq s_0$, $C \upharpoonright c = C[s] \upharpoonright c$, since

C is computably enumerable. Eventually the lower bound on the choice of traces for requirements grows beyond c. If c has not already been chosen as a trace by this stage, then $c \notin C$. So we suppose c is chosen at some stage s as the kth trace of some strategy α with requirement M_Φ assigned to it. Because $(\Phi(A \oplus C; k)\downarrow = \Phi(B \oplus C; k)\downarrow)[s]$, we can set $\Gamma(A \oplus B; c)[s] = 0$ with use $s \geq \max(\{\phi(A \oplus C; k)[s], \phi(B \oplus C; k)[s]\})$. If A never changes on either $\phi(A \oplus C; k)[s]$ or any $\phi^\beta(A \oplus C; l)$ for some $\beta^\frown\infty \subseteq \alpha$ and $c^\beta(l)\downarrow[s]$, and B never changes on $\phi(B \oplus C; k)[s]$ or a similar $\phi^\beta(A \oplus C; l)$, then c will never enter C, and we can continue to reset the use to this value until A and B stop changing below s. Otherwise, suppose one of these sets changes at some stage $s_1 > s$. If there is no restraint preventing c from entering C at stage $s_1 + 1$, then c will do so, and we can reset $\Gamma(A \oplus B; c)[s_1 + 1] = 1$ permanently with use s_1. If there is such a restraint, then it belongs to some β with authority to restrain c at s. There are only finitely many such nodes, since only finitely many nodes can have acted before stage s. By Lemma 4.4, if a node holding c at some $s' \geq s_1$ ever ceases to restrain c from entering C at $s'' > s'$, then some prior node must hold c at s''. In other words, at each subsequent stage when some node that once held c gives up restraint, the set of nodes which can possibly hold c at such a stage is reduced by one, since each of them must be prior to all the nodes which have previously held c. Eventually, there are either no more nodes which can hold c, in which case c must enter C, or some node holding c restrains c from C and never gives up its restraint on c. We therefore only need to show that for any node which holds c at some stage $s' \geq s_1$, we can determine from $A \oplus B$ whether or not the restraint imposed by such a node will ever be given up. By section 3.3 and Lemma 4.6, if such a node is never initialized, and A and B never change on its restraint keeping c from C, then c will never enter C. Lemma 4.6 applies here since $C \restriction c = C[s] \restriction c$. If we determine that the restraint is not permanent then we simply wait for the first stage at which either of these happens, and transfer our attention to the prior node which must hold c at such a stage if c does not enter C.

Suppose β is the node restraining c which is subsequent to all other restraining nodes, that is, the node that holds c at s_1. We can clearly determine from $A \oplus B$ whether or not A or B will ever change on the β-restraint keeping c from C. If not, we only need to show that we can determine from $A \oplus B$ whether or not β will ever be initialized. Notice that there are only finitely many nodes β_0 which have acted before β sets its restraint, and it is only through some change in the strategies assigned to one of these nodes that

β can be initialized. This follows from Definition 3.5 and the fact that a node can never set restraints or appoint traces at the first stage at which it is accessible. All initialization is of course caused by changes involving actual restraints and traces, so there is no need to consider nodes which are merely ghosts of actual nodes. For type-N nodes β_0, the necessary changes can be explicitly tied to changes in $A \oplus B$, since initialization will only occur when some restraint belonging to β_0 drops because of a change in $A \oplus B$, and the set of all such restraints is bounded at s_1. For type-M nodes β_0, the situation is slightly more complex. Because β has authority over c, by Definition 3.2, there must have been a β-stage s^β before s, the stage at which c was first chosen, such that β has never been initialized since stage s^β. By section 3.3, the initialization of β can only be caused by β_0 when some trace belonging to β_0 at s^β either enters or wishes to enter C. Since any such trace is less than c, the choice of s_0 implies that in fact such an initialization can only occur because such a trace first wishes to enter C at some stage. This can only happen when some computation existing at s^β becomes undefined because of some change in A or B below $\phi^{\beta_0}(l)[s^\beta]$ for some number l with $c^{\beta_0}(l)\downarrow[s^\beta]$. Hence, we can determine from $A \oplus B$ whether or not β will ever be initialized. As discussed above, this means we can eventually determine from $A \oplus B$ whether or not c will ever enter C.

References

[1] Ambos-Spies, K., *On pairs of recursively enumerable degrees*, Trans. Am. Math. Soc. 283 (1984), 507-531.

[2] R. Downey, *The $0'''$ priority method with special attention to density results*, Recursion Theory Week: Proceedings, Oberwohlfach 1989 (K. Ambos-Spies, *et al.* , eds. , Springer, Berlin, 1990).

[3] Fejer, P., *The density of the nonbranching degrees*, Ann. Pure Appl. Logic 24 (1983), 113-130.

[4] Kleene, S. C. and Post, E.L., *The upper semi-lattice of degrees of unsolvability*, Ann. of Math. (2) 59 (1954), 370-407.

[5] Lachlan, A., *Lower bounds for pairs of recursively enumerable degrees*, Proc. London Math. Soc. (3) 16 (1966), 537-569.

[6] Lachlan, A., *A recursively enumerable degree which will not split over all lesser ones*, Ann. Math. Logic 9 (1975), 307-365.

[7] Lachlan, A., *Bounding minimal pairs*, J. Symbolic Logic 44 (1979) 626-642.

[8] Lachlan, A., *Decomposition of recursively enumerable degrees*, Proc. Am. Math. Soc. 79 (1980), 629-634.

[9] Sacks, G., *On the degrees less than* $\mathbf{0}'$ Ann. of Math. (2) 77 (1963).

[10] Sacks, G., *The recursively enumerable degrees are dense* Ann. of Math. (2) 80 (1964), 300-312.

[11] Shore, R.A., *A non-inversion theorem for the jump operator*, Ann. Pure and Appl. Logic 40 (1988) 277-303.

[12] Shore, R.A and Slaman, T.A., *Working below a* low_2 *recursively enumerable degree*, Arch. Math. Logic 29 (1990), 201-211

[13] Slaman, T.A., *The density of infima in the recursively enumerable degrees*, Annals of Pure and Applied Logic, 52 (1991), 155-179.

[14] Spector, C., *On degrees of recursive unsolvability*, Ann. Math. 64 (1956), 581-592.

[15] Soare, R., *Recursively enumerable sets and degrees* (Springer, Berlin, 1987).

[16] Yates, C.E.M., *A minimal pair of recursively enumerable degrees*, J. Symbolic Logic 31 (1966), 159-168.