Settling Time Reducibility Orderings

by

Clinton Loo

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Pure Mathematics

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

It is known that orderings can be formed with settling time domination and strong settling time domination as relations on c.e. sets. However, it has been shown that no such ordering can be formed when considering computation time domination as a relation on n-c.e. sets where $n \geq 3$. This will be extended to the case of 2-c.e. sets, showing that no ordering can be derived from computation time domination on n-c.e. sets when $n \geq 2$.

Additionally, we will observe properties of the orderings given by settling time domination and strong settling time domination on c.e. sets, respectively denoted as $\mathcal{E}_{st}$ and $\mathcal{E}_{sst}$. More specifically, it is already known that any countable partial ordering can be embedded into $\mathcal{E}_{st}$ and any linear ordering with no infinite ascending chains can be embedded into $\mathcal{E}_{sst}$. Continuing along this line, we will show that any finite partial ordering can be embedded into $\mathcal{E}_{sst}$.

Acknowledgements

This thesis would not have been possible without all the pats on the back and slaps upside my head. I cannot even begin to express how grateful I am for all of those that have helped me along the way.

I would like to thank my supervisor, Barbara Csima. I could not have made it this far without all of her guidance, support, and long distance phone calls.

I would also like to thank Bernard Anderson and Johanna Franklin for their weekends spent in front of a chalkboard trudging through my work.

To Mom

Contents

Chapter 1

Introduction

In this thesis, two main results will be presented. The first will be an extension to the case $n = 2$ of an argument made in [2] which shows that the computation time domination relation does not result in a well defined ordering on n-c.e. sets where $n \geq 3$. The second gives that any finite partial ordering can be embedded into the ordering derived from the strong settling time domination relation on c.e. sets.

Both of these results have their roots in the settling time domination relation and its ordering on c.e. sets. The notion of settling time domination initially came about for the sake of Nabutovsky and Weinberger's work in differential geometry [5]. Essentially, settling time domination provides a means of comparing the complexities of the settling times of c.e. sets. Note that the notation and terminology appearing in this thesis correspond to those in Cooper's *Computability Theory* [1] and Soare's *Recursively Enumerable Sets and Degrees* [6].

Definition 1.1. For a c.e. set A with an associated enumeration $\{A_s\}_{s\in\omega}$, the settling (modulus) function m_A is defined by the following:

$$m_A(x) = (\mu s)[A_s \upharpoonright x = A \upharpoonright x]$$

where $A \upharpoonright x = \{y \in A \mid y \leq x\}$. That is, $m_A(x)$ gives the least stage such that the approximation settles up to x.

We can compare the complexities of the settling times of approximations of c.e. sets by extending the concept of domination.

Definition 1.2. A function f is said to be dominant if for each computable function g, for all but finitely many x, $f(x) > g(x)$. In particular, a function f is said to dominate a function g if for all but finitely many x, $f(x) > g(x)$.

Definition 1.3. For c.e. sets A and B with fixed enumerations $\{A_s\}_{s\in\omega}$ and $\{B_s\}_{s\in\omega}$, $\{A_s\}_{s\in\omega}$ is said to settling time dominate $\{B_s\}_{s\in\omega}$ if for each computable function f, for all but finitely many arguments x, $m_A(x) > f \circ m_B(x)$. When the fixed enumerations of A and B are understood, this is denoted by $A >_{st} B$.

Intuitively, $A >_{st} B$ means that the approximation of A settles so much slower than the settling time of B's approximation, that there is no computable means of slowing the settling times of B's approximation to those of A's approximation. It has been shown by Nies (see [4]) that settling time domination is actually independent of the chosen approximations of the c.e. sets. This being the case, the relation $>_{st}$ induces a relation on c.e. sets. We use the same notation for this new relation on c.e. sets, and write $A >_{st} B$ to denote that the c.e. set A settling time dominates the c.e. set B. Furthermore, we denote by $\mathcal{E}_{st}$ the structure of c.e. sets with the relation $<_{st}$. Naturally, it was asked whether such an ordering could be extended to Δ^0_2 sets.

Definition 1.4. A set is said to be Δ^0_2 if it is limit computable. That is, A is Δ^0_2 if there is a computable sequence $\{A_s\}_{s\in\omega}$ of finite sets such that for all x, $\lim_s A_s(x)$ exists and $\lim_s\ A_s(x) = A(x)$.

In general, it was shown that such an ordering could not be derived from settling time domination with regards to Δ^0_2 sets [2]. Indeed, it fails for the following subset of Δ^0_2 sets.

Definition 1.5. A set A is said to be n-c.e. if there is a computable sequence $\{A_s\}_{s\in\omega}$ such that $A_0 = \emptyset$, $A_s(x) \in \{0, 1\}$, $\lim_s A_s = A$, and for any given x,

$$|\{s : A_s(x) \neq A_{s+1}(x)\}| \leq n.$$

In particular, the sequence $\{A_s\}_{s\in\omega}$ is said to be an n-c.e. approximation, and A is said to be properly n-c.e. if it is n-c.e. and not m-c.e. for all $m < n$. Essentially, an n-c.e. set is a set with a computable approximation such that any number can be enumerated and removed from the approximation at most n times.

To be specific, it is shown in [2] that settling time domination is not well defined with respect to n-c.e. sets for $n \geq 2$. Since an ordering comparing the complexities of the approximations of Δ^0_2 sets could not be constructed based on their settling time, it was then natural to try to see if this could be done with regards to computation time.

Definition 1.6. For a Δ^0_2 set A with an approximation $\{A_s\}_{s\in\omega}$, the computation function C_A is defined as

$$C_A(x) = (\mu s \geq x)[A_s \upharpoonright x = A \upharpoonright x].$$

In other words, $C_A(x)$ is the first stage greater than or equal to x such that the approximation appears correct up to the number x.

To try to compare the computation times of Δ^0_2 sets, the following relation is used which is basically the same as the notion of settling time domination but with the computation function in the place of the settling function.

Definition 1.7. Let A and B be Δ^0_2 sets with respective approximations $\{A_s\}_{s\in\omega}$ and $\{B_s\}_{s\in\omega}$. The approximation $\{A_s\}_{s\in\omega}$ is said to computation time dominate the approximation $\{B_s\}_{s\in\omega}$ if for each computable function, f, for all but finitely many arguments x, $C_A(x) > f \circ C_B(x)$.

Note that with regards to a Δ^0_2 set, A, the settling function is defined as

$$m_A(x) = (\mu s)(\forall t \geq s)[A_t \upharpoonright x = A \upharpoonright x].$$

Essentially, when dealing with Δ^0_2 sets the computation function gives the first stage at which the approximation appears correct on an initial segment, while the settling function gives the first stage at which the approximation appears correct and remains constant on an initial segment. In general, these stages will differ. However, when dealing with c.e. sets these two stages coincide and as such, the computation function and settling function behave similarly. With Δ^0_2 sets, it is quite natural to use the computation function rather than the settling function as Δ^0_2 approximations generally do not have the simplicity of the approximations of c.e. sets. Although the computation function and settling function were essentially the same with regards to c.e. sets, in general they behave differently. Indeed,

the computation function of a Δ^0_2 set A is Turing equivalent to A but the same cannot necessarily be said of the settling function of A [1].

Even with the computation function used in the place of the settling function the relation still failed to be well defined with regards to n-c.e. sets where $n \geq 3$ [2]. However, the question remained whether it was possible for the computation time domination relation to order the particular case of the 2-c.e. sets even though it failed in general with regards to Δ^0_2 sets. In Chapter 3, it is shown that an ordering will also fail to come about with 2-c.e. sets, essentially, closing the door on trying to generalize some notion of a settling time domination ordering to n-c.e. sets.

On the other end of the spectrum is the strong settling time domination relation. Indeed, whereas computation time domination came about from trying to extend the notion of settling time domination to Δ^0_2 sets, the strong settling time domination relation is a subset of the settling time domination relation with regards to c.e. sets.

Definition 1.8. For c.e. sets A and B with respective enumerations $\{A_s\}_{s\in\omega}$ and $\{B_s\}_{s\in\omega}$, A is said to strongly settling time dominate B if for all computable functions f and g, for all but finitely many arguments x, $m_A(x) > f \circ m_B \circ g(x)$. This is denoted by $A >_{sst} B$. Further, we denote by $\mathcal{E}_{sst}$ the structure of c.e. sets with the relation $<_{sst}$.

It is shown in [3] that any computable partial ordering can be embedded into the structure $\mathcal{E}_{st}$. It remains unknown whether or not such an embedding is possible for the strong settling time domination ordering. However, it has been shown that any computable linear ordering with no infinite ascending sequences can be embedded into the strong settling time domination ordering [3]. Continuing along this line, it is shown in Chapter 4 that any finite partial ordering can also be embedded into $\mathcal{E}_{sst}$.

Before proceeding, some basic conventions and notation that are used throughout this thesis are given. It is assumed that for a partial computable function, φ_e, if $\varphi_{e,s}(x) \downarrow$ then for all $y < x$, $\varphi_{e,s}(y) \downarrow$ and $s > \varphi_e(x), x, e$. Further, we assume that for all $m < n$ that $\varphi_e(m) \leq \varphi_e(n)$ if both $\varphi_e(m)$ and $\varphi_e(n)$ converge. That is, we use an enumeration of the non-decreasing partial computable functions with initial segments of ω as domains. Notice that this will still suffice although the definitions of settling time domination and strong settling time domination call for any computable function. Indeed, any computable function will have all of ω as a domain and as such any statement made regarding all or almost

all partial computable functions with an initial segment of ω as a domain will hold for the computable functions. Moreover, the requirements of settling time domination and strong settling time domination will still hold when regarding just the non-decreasing partial computable functions with initial segments of ω as domains. Suppose φ is a partial computable function with an initial segment of ω as a domain but not necessarily non-decreasing and consider the function φ^* defined by $\varphi^*(0) = \varphi(0)$ and $\varphi^*(x) = \max\{\varphi(x), \varphi^*(x-1)\}$ for $x > 0$. Now φ^* is non-decreasing and for all x if $\varphi(x) \downarrow$ then $\varphi^*(x) \geq \varphi(x)$. Thus, for c.e. sets A and B, if $m_A(x) > \varphi^* \circ m_B(x)$ then $m_A(x) > \varphi^* \circ m_B(x) \geq \varphi \circ m_B(x)$ and so, $m_A(x) > \varphi \circ m_B(x)$ as we desire.

Also, throughout this paper by partial ordering we mean strict partial ordering. Lastly, we write $\forall^\infty x$ to denote *for all but finitely many* x and $\exists^\infty x$ to denote *there exists infinitely many* x.

Chapter 2

Settling Time Domination

As mentioned in the previous chapter, settling time domination provides a means of comparing the complexity of the settling times of approximations of c.e. sets. Indeed, the approximation $\{A_s\}_{s\in\omega}$ is said to settling time dominate the approximation $\{B_s\}_{s\in\omega}$, $A >_{st} B$, if $\{A_s\}_{s\in\omega}$ settles so much slower than $\{B_s\}_{s\in\omega}$ that there is no computable means of raising the settling times of $\{B_s\}_{s\in\omega}$ to those of $\{A_s\}_{s\in\omega}$. In this chapter several theorems regarding settling time domination and the structure $\mathcal{E}_{st}$ are given. In addition, an example demonstrating the techniques used to embed a partial ordering into $\mathcal{E}_{st}$ is provided.

First, we show that settling time domination is independent of the chosen approximations of the c.e. sets. That is, the ordering in $\mathcal{E}_{st}$ is well defined.

Theorem 2.1 (Nies, see [4])**.** *If $\{\hat{A}_s\}_{s\in\omega}$ and $\{\tilde{A}_s\}_{s\in\omega}$ are two c.e. approximations of the set A and $\{\hat{B}_s\}_{s\in\omega}$ and $\{\tilde{B}_s\}_{s\in\omega}$ are two c.e. approximations of the set B and $\hat{A} <_{st} \hat{B}$ then $\tilde{A} <_{st} \tilde{B}$.*

Proof. Before proceeding we will need the following lemma.

Lemma 2.2. *Given two c.e. approximations, $\{\hat{A}_s\}_{s\in\omega}$ and $\{\tilde{A}_s\}_{s\in\omega}$, of a set A there exists a computable function f such that for all x, $f \circ m_{\tilde{A}}(x) \geq m_{\hat{A}}(x)$. Indeed, there exists a strictly increasing computable function f' such that $f' \circ m_{\tilde{A}}(x) \geq m_{\hat{A}}(x)$.*

Proof. First let $f(0) = 0$. For $n > 0$, let $s_n > n$ be the least stage such that there is some number y that enters $\tilde{A}$. Now let $t_n > s_n$ be the least stage such that $\tilde{A}_{t_n} \upharpoonright y = \hat{A}_{t_n} \upharpoonright y$.

Such a t_n must exist, as $\{\hat{A}_s\}_{s\in\omega}$ and $\{\tilde{A}_s\}_{s\in\omega}$ approximate the same set. Define $f(n) = t_n$ for all $n > 0$. This function meets the requirements of the lemma.

Verification: Suppose $n = m_{\tilde{A}}(x)$ for some x. Note that as $n = m_{\tilde{A}}(x)$ and $s_n > n$, the element, y, that enters $\tilde{A}$ at stage s_n must be greater than x. Since $\{\tilde{A}_s\}_{s\in\omega}$ is a c.e. approximation, $x < y$, and $f(n) = t_n > n$ it is the case that $\tilde{A}_n \restriction x \subset \tilde{A}_{f(n)} \restriction y$. Then as $n = m_{\tilde{A}}(x)$, we have that $A \restriction x \subset \tilde{A}_{f(n)} \restriction y$. So the fact that $\tilde{A}_{f(n)} \restriction y = \hat{A}_{f(n)} \restriction y$ gives $A \restriction x \subset \hat{A}_{f(n)} \restriction y$ and thus, $f(n) \geq m_{\hat{A}}(x)$. Hence, $f \circ m_{\tilde{A}}(x) \geq m_{\hat{A}}(x)$ as desired.

To obtain a strictly increasing computable function satisfying the requirements above, define $f'(0) := f(0)$ and $f'(n) := \max\{f(n), f'(n-1)+1\}$ for $n > 0$.

□

Now Lemma 2.2 gives strictly increasing computable functions f and g such that:

1. For all x, $f \circ m_{\tilde{B}}(x) \geq m_{\hat{B}}(x)$
2. For all x, $g \circ m_{\hat{A}}(x) \geq m_{\tilde{A}}(x)$.

Let h be a computable function. Our conventions give that h is non-decreasing and so from (2) we have that $(\forall x)[h \circ g \circ m_{\hat{A}}(x) \geq h \circ m_{\tilde{A}}(x)]$. Since f is strictly increasing we get,

$$(\forall x)[f \circ h \circ g \circ m_{\hat{A}}(x) \geq f \circ h \circ m_{\tilde{A}}(x)].$$

Now as f, g, and h are computable so is $f \circ h \circ g$. From our assumption that $\hat{B} >_{st} \hat{A}$, it follows that for all but finitely many x, $m_{\hat{B}}(x) > f \circ h \circ g \circ m_{\hat{A}}(x)$. So then (1) gives that $(\forall^\infty x)[f \circ m_{\tilde{B}}(x) > f \circ h \circ m_{\tilde{A}}(x)]$. Lastly, from f's monotonicity we have $(\forall^\infty x)[m_{\tilde{B}}(x) > h \circ m_{\tilde{A}}(x)]$. Hence, $\tilde{B} >_{st} \tilde{A}$.

□

This gives that $<_{st}$ is well defined on c.e. sets. In other words, settling time domination is independent of the particular c.e. enumerations chosen. Thus, we have that $<_{st}$ is, indeed, an ordering on c.e. sets and as such, it actually makes *sense* to study the structure, $\mathcal{E}_{st}$.

Theorem 2.3 (Csima, Shore [3])**.** *Any countable partial ordering can be embedded into* $\mathcal{E}_{st}$*.*

The following is an example demonstrating the techniques, used in Theorem 2.3, to embed a partial ordering into $\mathcal{E}_{st}$. In particular, it demonstrates how to build c.e. sets incomparable in the $>_{st}$ ordering (a technique which is later needed in Chapter 4).

Example 2.4. The partial ordering P defined by

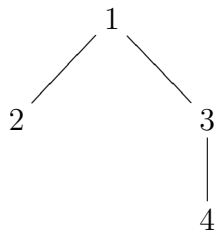

Figure 2.1

can be embedded into $\mathcal{E}_{st}$. That is, we can build c.e. sets A_1, A_2, A_3, and A_4 such that $A_p >_{st} A_q$ if and only if $p >_P q$.

Proof. We build these sets stage by stage ensuring that:

$$R_{p>_P q} : \text{If } p >_P q \text{ then for each total } f, \text{ we have } (\forall^\infty x)[m_{A_p}(x) > f \circ m_{A_q}(x)]$$
$$R_{p\not>_P q} : \text{If } p \not>_P q \text{ then } (\exists^\infty x)[m_{A_q}(x) \geq m_{A_p}(x)].$$

For the sake of the requirement $R_{p>_P q}$, if $p >_P q$ we have elements that *guard* A_p's settling time domination of A_q. To do this, the commitment is made that a possible entrant x is appointed to A_q at some stage s only if there are appointed *guards* $x_0, x_1, \ldots,$ and x_s less than x such that the guard x_i will enter A_p at the stage t when $\varphi_i(s)\downarrow$. Recall from our conventions that $t > \varphi_i(s)$ and so, we will get that $m_{A_p}(x) > \varphi_i \circ m_{A_q}(x)$.

For the sake of the requirement $R_{p\not>_P q}$, if $p \not>_P q$ then at each stage s an element n^q_s is chosen and enumerated into A_q with the intention of enumerating an element less than n^q_s into A_q at any stage $t \geq s$ such that an element less than or equal to n^q_s is enumerated

into A_p. From these actions we get that $m_{A_q}(n_s^q) \geq m_{A_p}(n_s^q)$ and since we choose an n_s^q for every stage s we, indeed, have that $(\exists^\infty x)[m_{A_q}(x) \geq m_{A_p}(x)]$.

Following the above plan, it is the case that at any given stage there are only finitely many elements entering or committed to enter any given A_p. Further note that since $A_2 \not>_{st} A_4$ and $A_3 >_{st} A_4$ we get $A_2 \not>_{st} A_3$ for free, otherwise we would have $A_2 >_{st} A_3 >_{st} A_4$ and hence, $A_2 >_{st} A_4$. That said, to preserve the incomparability in the partial ordering it suffices to just build $A_2,\ A_3$, and A_4 such that $A_2 \mid_{st} A_4$, $A_3 \not>_{st} A_2$. To preserve the relation $>_P$, we build $A_1,\ A_2,\ A_3$, and A_4 such that $A_1 >_{st} A_2$, $A_1 >_{st} A_3$, and $A_3 >_{st} A_4$. We now proceed on to the construction.

Construction:

Stage s: We will first choose an n_s^2 and then, in turn, an n_s^4. Note that when choosing n_s^2 we must keep in mind that we have to build A_2 and A_1 so that $A_2 <_{st} A_1$. To ensure that $A_2 <_{st} A_1$, for each possible entrant x appointed to A_2 in this current stage we will choose numbers $x_0, \ldots,\ x_s < x$ with the commitment that x_i will enter A_1 at the stage when $\varphi_i(t) \downarrow$ where t denotes the stage when x enters A_2. Additionally, to account for n_s^2, itself, we will choose $(s+1)$-many guards less than n_s^2 such that the i^{th} guard will enter A_1 at the stage when $\varphi_i(s) \downarrow$. Now n_s^2 will be chosen to allow room for all of these guards and with the intention that whenever an element less than it enters A_3 or A_4, there will be some number less than n_s^2 that enters A_2.

Similarly, n_s^4 will be chosen with the chain $A_4 <_{st} A_3 <_{st} A_1$ in mind. That is, for each possible entrant y appointed to A_4 at this stage there must be $(s+1)$-many guards $y_0, \ldots,\ y_s$ less than y with the commitment that the guard y_i enters A_3 at the stage when $\varphi_i(t') \downarrow$ where t' denotes the stage when y enters A_4. Additionally, there will be $(s+1)$-many guards less than n_s^4 such that the i^{th} guard enters A_3 at the stage when $\varphi_i(s)$ converges. In turn, for each possible entrant z appointed to A_3 at this stage we must have guards $z_0, \ldots,$ and z_s such that the i^{th} guard enters A_4 at the stage when the partial computable function φ_i converges with the stage when z enters A_3 as its argument. Now n_s^4 will be chosen with the intention that whenever a number less than it enters A_2, there will be a number less than n_s^4 that enters A_4 while also allowing enough room for all of the necessary guards of A_3 and A_4.

Before proceeding further note that choosing the incomparability markers in this particular order is not necessary. Indeed, an n_s^4 could be chosen first and then an n_s^2.

We now choose the incomparability markers n_s^2 and n_s^4. Since there are only a finite number of possible entrants appointed to A_3 and A_4 from previous stages and thus far in this current stage, say k many, we may choose n_s^2 and guards $x_1,\ x_2, \ldots,\ x_k$ to be greater than any number discussed thus far in the procedure. Furthermore, we may choose n_s^2 and corresponding guards that are spread out enough to provide enough room for the guards that are needed for the sake of $A_2 <_{st} A_1$ and the functions $\varphi_0, \ldots,\ \varphi_s$. The n_s^2 guard x_i has the commitment of entering A_2 when the i^{th} possible entrant less than or equal to n_s^2 enters A_3 or A_4. Now enumerate n_s^2 into A_2.

We now choose an n_s^4. As only finitely many elements are appointed to enter A_2 from commitments made from previous stages and this current stage, say j many such elements, we may choose an n_s^4 and elements $y_1,\ y_2, \ldots,\ y_j < n_s^4$ greater than any number discussed thus far and spread out enough to provide room for guards for the partial computable functions $\varphi_0, \ldots, \varphi_s$ and for the sake of both $A_4 <_{st} A_3$ and $A_3 <_{st} A_1$. These y_i have the commitment of entering A_4 when the i^{th} possible entrant less than or equal to n_s^4 enters A_2. Now enumerate n_s^4 into A_4.

To ensure that $A_2 <_{st} A_1$, choose for each possible entrant, x, appointed to A_2 at this stage numbers $x_0', \ldots, x_s' < x$ such that x_i' enters A_1 at the stage when $\varphi_i(t) \downarrow$ where t denotes the stage when x enters A_2. To account for the marker n_s^2, choose for $(s+1)$-many guards less than n_s^2 so that the i^{th} guard enters A_1 at the stage when $\varphi_i(s) \downarrow$. For the sake of $A_4 <_{st} A_3$ we perform similar actions. Choose for each possible entrant, z, appointed to A_4 at this stage (there are only finitely many), numbers $z_0, \ldots, z_s < z$ such that z_i enters A_3 at the stage when $\varphi_i(u) \downarrow$ where u denotes the stage when z is enumerated into A_4. Additionally, to account for n_s^4, choose $(s+1)$-many guards less than n_s^4 such that the i^{th} guard enters A_3 at the stage when $\varphi_i(s) \downarrow$. Similarly, for the sake of $A_3 <_{st} A_1$, choose for each possible entrant, w, appointed to A_3 at this stage $(s+1)$-many guards such that the i^{th} guard enters A_1 at the stage when $\varphi_i(v)$ converges where v denotes the stage when w enters A_3. Notice that we may do this as the y_i from the previous paragraph were chosen to be spread out enough to provide room for these guards.

Note that when each n_s^p is chosen, it and its guards are chosen to be greater than any number discussed up to that point. So then we have that these markers and their guards will not be less than previously appointed markers and guards. That being the case, they will not affect the number of possible entrants previously appointed. As such, we have only appointed finitely many possible entrants to enter any given A_p. Moreover, choosing

the n^p_s in such a manner will prevent any previous commitments from being undermined. That is, the number of possible entrants less than a given n^p_s will not increase after the n^p_s is chosen and as such, it has an appropriate guard accounting for each of these possible entrants.

Notice that at any point in the above construction there are only finitely many possible entrants being discussed and only finitely many functions are considered (for example, at stage s we will only consider $\varphi_0, \ldots, \varphi_s$). That said, when we choose an n^p_s we can always compute how many guards will be needed to preserve the ordering of P and choose n^p_s and its guards accordingly. This guarantees that we can always choose our markers spaced far enough apart so as to provide enough room for the required number of guards.

Lemma 2.5. *If $p >_P q$ then $A_p >_{st} A_q$.*

Proof. Suppose f is a total function and suppose that i is the index such that $f = \varphi_i$. Let $s \geq i$ be the stage such that every entrant that was appointed to enter A_q prior to stage i that will actually enter A_q has already done so. That being the case, any x that enters A_q after stage s was appointed to enter A_q after stage i. So then φ_i is considered when x is appointed and an $x_i < x$ is appointed to enter A_p at the stage such that $\varphi_i(t) \downarrow$ where t is the stage when x enters A_q. That all said, we have that for all x entering A_q after stage s, $m_{A_p}(x) > \varphi_i \circ m_{A_q}(x)$. Now as only finitely many numbers could have entered A_q prior to stage s, we have $(\forall^\infty x)[m_{A_p}(x) > \varphi_i \circ m_{A_q}(x)]$ as was desired.

□

Lemma 2.6. *If $p \not>_P q$ then $A_p \not>_{st} A_q$.*

Proof. At each stage s an n^q_s is enumerated into A_q and for each $x < n^q_s$ enumerated into A_p after stage s, there is an element $y < n^q_s$, namely one of the guards belonging to n^q_s, appointed to enter A_q at the same stage. This gives that $m_{A_q}(n^q_s) \geq m_{A_p}(n^q_s)$. Hence, there are infinitely many x, namely the n^q_s for $s \in \omega$, such that $m_{A_q}(x) \geq m_{A_p}(x)$ and so $A_p \not>_{st} A_q$.

□

Thus, A_1, A_2, A_3, and A_4 are as desired and as such, the partial ordering P can be embedded into $\mathcal{E}_{st}$.

□

Naturally, one would ask if an analogous statement of Theorem 2.3 would hold in the strong settling time domination setting. Although, the question remains open we will take a step in attempting to answer it in the positive by showing in Chapter 4 that any finite partial order can be embedded in $\mathcal{E}_{sst}$.

In the following chapter, we see that although the settling time domination relation can order the c.e. sets, it will, in general, fail as an order on Δ_2^0 sets. In particular, the relation $>_{st}$ fails to be well defined on n-c.e. sets for $n \geq 2$. It is also seen that the relation fails to be well defined on these sets even when considering computation times rather than settling times.

Chapter 3

Computation Time Domination

Given how well behaved the settling time domination relation is on c.e. sets, it is natural to see if it could be lifted to Δ^0_2 sets. Unfortunately, the approximation independence from Theorem 2.1 does not generalize to n-c.e. sets and their n-c.e. approximations when $n \geq 2$. Indeed, we have the following result.

Theorem 3.1 (Csima [2])**.** *For any $n \geq 2$ there exists a properly n-c.e. set A with two n-c.e. approximations $\{A_s\}_{s\in\omega}$ and $\{\tilde{A}_s\}_{s\in\omega}$ such that for any computable function f, for all but finitely many x, $m_{\tilde{A}}(x) \geq f \circ m_A(x)$. In fact, such an A can be found in every proper n-c.e. degree.*

Note that when $n = 1$, an n-c.e. set is simply a c.e. set and so approximation independence holds. However, unlike c.e. sets the computation times of Δ^0_2 sets do not necessarily coincide with their settling times. That being the case, even though an ordering fails to come about with regards to the settling times of Δ^0_2 sets, one could try comparing Δ^0_2 sets with regards to the computation times of their approximations.

In this chapter we discuss computation time domination. As mentioned, the use of the computation function is much more fitting when considering Δ^0_2 sets rather than c.e. sets which have relatively simplistic approximations. In particular, we consider computation time domination with regards to n-c.e. sets. As seen in Theorem 3.1 it is known that settling time domination is not well defined with regards to n-c.e. approximations of n-c.e. sets where $n \geq 2$. In this chapter we show that an analogous theorem holds when considering computation time domination. That is, when replacing the settling function

with the computation function an ordering of the Δ^0_2 sets still fails to come about. Indeed, even trying to order the Δ^0_2 sets by comparing their computation times still fails to be well defined with regards to n-c.e. sets for $n \geq 3$. This result is extended to the case when $n = 2$ which shows that not only does computation time domination fail to order Δ^0_2 sets, it fails for even the n-c.e. sets where $n \geq 2$. First note that when considering c.e. approximations of c.e. sets, the computation function behaves much like the settling function and as such computation time domination is well defined with regards to c.e. approximations of c.e. sets.

Theorem 3.2 (Csima [2])**.** *There exists a c.e. set A with a 3-c.e. approximation $\{\tilde{A}_s\}_{s\in\omega}$ and a c.e. approximation $\{A_s\}_{s\in\omega}$ such that for any computable function f, for all but finitely many x, $C_A(x) > f \circ C_{\tilde{A}}(x)$.*

The key technique used to build the approximations in Theorem 3.2 is the idea of a *temptation*. A temptation is performed by enumerating and then subsequently removing an element from one of the approximations. As these are 3-c.e. approximations one may still enumerate these numbers back into both sets. The key point of a temptation is to provide one approximation with an earlier configuration that may be returned to by re-enumerating elements into the approximation. Now when these elements are enumerated into both sets, the other approximation will not have seen this configuration before and as such this approximation will have higher computation values than those of the approximation returning to an old configuration.

As a basic example illustrating the techniques used to build the approximations in Theorem 3.2 we build A and $\tilde{A}$ with only the number 0 and the partial computable function φ_0 in mind. Indeed, we build A and $\tilde{A}$ so that $C_A(0) > \varphi_0 \circ C_{\tilde{A}}(0)$. At stage s, enumerate 0 into $\tilde{A}$ and subsequently remove it from $\tilde{A}$ at stage $s+1$. Note that A is not acted upon during the temptation. At the stage t that $\varphi_0(s+1)\downarrow$, enumerate 0 into both sets. As 0 had already been enumerated into $\tilde{A}$ at stage s we have that $C_{\tilde{A}}(0) = s$. On the other hand, A is not acted upon at all until 0 was enumerated into it at stage t. Thus, we have that

$$C_A(0) = t > \varphi_0(s+1) \geq \varphi_0(s) = \varphi_0 \circ C_{\tilde{A}}(0),$$

where $\varphi_0(s+1) \geq \varphi_0(s)$ holds by our convention that the partial computable functions are non-decreasing.

In fact, we have that Theorem 3.2 extends to properly n-c.e. sets when $n \geq 3$. That is, rather than just observing a 3-c.e. approximation of a c.e. set we have that for any $n \geq 3$, there exists a properly n-c.e. set with two n-c.e. approximations such that one computation time dominates the other. Using the same techniques from 3.2 while also coding in a properly n-c.e. set D gives the following.

Theorem 3.3 (Csima [2])**.** *For any $n \geq 3$ and any properly n-c.e. set D there is an n-c.e. set $A \geq_m D$ where A has two n-c.e. approximations $\{A_s\}_{s\in\omega}$ and $\{\tilde{A}_s\}_{s\in\omega}$ such that for any computable function f, for all but finitely many x, $C_A(x) > f \circ C_{\tilde{A}}(x)$.*

The technique of using a temptation to build 3-c.e. approximations such that one computation time dominates the other does not, however, carry over to 2-c.e. approximations. The most obvious problem is that elements are enumerated, removed, and then re-enumerated, which cannot happen in a 2-c.e. approximation. One may think then to just enumerate into an approximation and then remove based on the convergence of the partial computable functions. Although we are doing such actions in the proof of the following theorem, our techniques stray from those used for 3-c.e. sets, in that we must perform similar actions for the other approximation as well. Indeed, if the other approximation is left alone as it was in the temptation process, what would we do when the partial computable functions converged? If we enumerated into one approximation, while removing elements from the other, there would be no guarantee that they would end up approximating the same set. As shown in the following theorem, there is an enumeration process such that elements are enumerated into both approximations and elements are removed from both approximations due to the convergence of the partial computable functions.

Theorem 3.4. *There exists a 2-c.e. set A with two 2-c.e. approximations, $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$, such that for any computable function f, for all but finitely many x, $C_{\hat{A}}(x) > f \circ C_{\tilde{A}}(x)$. That is, computation time domination is not approximation-independent when considering only 2-c.e. approximations of 2-c.e. sets.*

Proof. To meet the requirements of the theorem, we seek to build $\hat{A}$ and $\tilde{A}$ stage by stage using an enumeration process on blocks of consecutive numbers that results in any initial segment of $\tilde{A}$ eventually settling on a configuration that appears in an earlier stage during the construction and the initial segment of $\hat{A}$ settling on the same configuration, but unlike the case with $\tilde{A}$, this configuration is entirely new to $\hat{A}$. To do this, for some

given block, the numbers in the block are enumerated into one approximation in ascending order and in descending order for the other and any element that is removed is done so simultaneously from both approximations. Thus, any initial segment can only have differing configurations in the two approximations for finitely many stages and as such, the approximations eventually match on the initial segment and no longer change. Indeed, as $\tilde{A}$ returns to an old configuration and $\hat{A}$ has never seen this configuration before, the computation times of $\hat{A}$ are higher than those of $\tilde{A}$.

Now, we must have that the computation times of $\hat{A}$ are so much higher than those of $\tilde{A}$, that $\hat{A}$ actually computation time dominates $\tilde{A}$. To achieve this, any element that is removed is done so based on the convergence of a partial computable function with the stage at which the enumeration process is completed on the block containing that number as an argument. So then any initial segment of $\tilde{A}$ settles on a configuration that appears during the enumeration process on some block containing members of the initial segment. However, the same initial segment in $\hat{A}$ does not settle on that configuration until a partial computable function converges with the last stage of that enumeration process as its argument.

Before proceeding to the main construction, a couple of basic examples illustrating the element enumeration and removal techniques of the construction are discussed. First, only the partial computable function φ_0 and the pair of numbers 0 and 1 are considered and we will show how to construct c.e. sets $\tilde{A}$ and $\hat{A}$ such that $C_{\hat{A}}(1) > \varphi_0 \circ C_{\tilde{A}}(1)$. Note that the configurations of the characteristic functions of $\tilde{A}$ and $\hat{A}$ in the following process appear as in Figure 3.1 on the next page.

At stage 1, enumerate 1 into $\tilde{A}$ and 0 into $\hat{A}$ and then at stage 2 enumerate 0 into $\tilde{A}$ and 1 into $\hat{A}$. That is, enumerate 0 and 1 in ascending order into $\hat{A}$ and in descending order into $\tilde{A}$. Now if $\varphi_0(2) \downarrow$ at a later stage t then remove 0 from both sets. As 0 and 1 were enumerated into $\tilde{A}$ in descending order, the configuration of $\tilde{A}$ at stage 1 matches the current one. As 0 and 1 were enumerated into $\hat{A}$ in ascending order, this current configuration is new to $\hat{A}$. Hence,

$$C_{\hat{A}}(1) = t > \varphi_0(2) \geq \varphi_0(1) = \varphi_0 \circ C_{\tilde{A}}(1)$$

where $\varphi_0(1) = \varphi_0 \circ C_{\tilde{A}}(1)$ holds since $C_{\tilde{A}}(1) = 1$.

The configurations of the characteristic functions of $\tilde{A}$ and $\hat{A}$ in the process outlined above appear as:

	$\tilde{A}$	$\hat{A}$
	01	01
Stage 0	00	00
Stage 1	01	10
Stage 2	11	11
$\vdots$	$\vdots$	$\vdots$
Stage t	01	01

Figure 3.1

As Figure 3.1 shows, $\tilde{A}$ has a configuration at stage 1 that matches the final configuration at stage t and $\hat{A}$ does not have such a configuration until stage t. Note that the enumeration process is not performed until stage 1 so any actions will affect the computation values of both 0 and 1.

Now in the case that $\varphi_0(2)$ does not converge, the configuration at stage 2 remains fixed in both approximations. That being the case, $\tilde{A}$ and $\hat{A}$ are equal regardless if φ_0 is total or not. Additionally, φ_0 does not affect $\hat{A}$'s computation time domination of $\tilde{A}$ since only total computable functions are taken into account.

Note that we do not get $C_{\hat{A}}(0) > f \circ C_{\tilde{A}}(0)$ from the process above. This may appear troublesome as domination must hold for almost every element but even when considering just 0 and 1 we already do not have domination for an element. However, as it will turn out, 0 is the only problematic element when considering just φ_0. To illustrate this, another example is given where φ_0 and the numbers from 0 to 3 are considered.

First divide the numbers into two blocks, B_0 and B_1, where B_0 consists of the numbers 0 and 1 and B_1 consists of the remaining numbers. The following process is quite similar to the previous procedure in that the elements of a given block are enumerated in ascending order into $\hat{A}$ and descending order into $\tilde{A}$ and any elements that are removed are done so based upon the convergence of φ_0. However, now that there are more elements taken into consideration, the procedure must also act for the block B_1 consisting of the numbers 2 and 3.

Much like the previous process we wait for $\varphi_0(s)$ to converge where the enumeration process for B_0 was completed at stage s and then remove 0 from both sets. It is after

this stage that we enumerate the elements of B_1 into both sets. Again, we follow our enumeration process and enumerate the elements of B_1 in ascending order into $\hat{A}$ and descending order into $\tilde{A}$. This enumeration provides $\tilde{A}$ with an old configuration that it returns to if any more elements are removed from B_1. Note that as B_0 is no longer acted upon, the configurations resulting from B_1's enumeration are not compromised.

At stage 3, perform the enumeration process for the elements of B_0, enumerating in ascending order into $\hat{A}$ and descending order into $\tilde{A}$, which completes at stage 4. Now at the stage t when $\varphi_0(4) \downarrow$ remove 0 from both sets. Following this stage, perform the enumeration process for the elements of B_1 which completes at stage $t+2$. At the stage u when $\varphi_0(t+2)$ converges remove the number 2 from both sets. As was the case in the previous procedure, if φ_0 does not converge then $\tilde{A}$ and $\hat{A}$ still appear the same and φ_0 will not affect $\hat{A}$'s computation time domination of $\tilde{A}$ as it is not total. Also note that as the procedure does not start until stage 3, all of the actions outlined in the process above affect the computation values in both blocks.

The configurations of the characteristic functions of $\tilde{A}$ and $\hat{A}$ during the process appear as:

	$\tilde{A}$		$\hat{A}$	
	B_0	B_1	B_0	B_1
	01	23	01	23
Stages $0-2$	00	00	00	00
Stage 3	01	00	10	00
Stage 4	11	00	11	00
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage t	01	00	01	00
Stage $t+1$	01	01	01	10
Stage $t+2$	01	11	01	11
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage u	01	01	01	01

Figure 3.2

As one can see from Figure 3.2, $\tilde{A}$ and $\hat{A}$ match at the final stage u. Furthermore, at stage t when 0 is removed from both sets, there is an earlier matching configuration of B_0 in $\tilde{A}$ at stage 3. Similarly at stage u when 2 is removed from both sets there is a matching configuration in $\tilde{A}$ at stage $t+1$. However, there are no earlier matching configurations in

$\hat{A}$. The argument for $C_{\hat{A}}(1) > \varphi_0 \circ C_{\tilde{A}}(1)$ is similar to the one in the previous example. Now the final configuration of the initial segment up to 3 matches the configuration in $\tilde{A}$ at stage $t+1$ and does not appear in $\hat{A}$ until the stage u when $\varphi_0(t+2)$ converges. So then,

$$\varphi_0 \circ C_{\tilde{A}}(3) = \varphi_0(t+1) \leq \varphi_0(t+2) < u = C_{\hat{A}}(3)$$

where $\varphi_0 \circ C_{\tilde{A}}(3) = \varphi_0(t+1)$ holds since $C_{\tilde{A}}(3) = t+1$. Lastly, consider the configurations of the initial segment up to the number 2. As seen earlier, the first element of B_0 is problematic since $\hat{A}$ fails to computation time dominate $\tilde{A}$ on the number 0 but there are no such issues with the first element of B_1. The final configuration of the initial segment up to 2 in Figure 3.2 at stage u is 010 which appears earlier in $\tilde{A}$ at stage 3 during the enumeration process for B_0 but does not appear in $\hat{A}$ until stage t when $\varphi_0(4)$ converges. That being the case,

$$\varphi_0 \circ C_{\tilde{A}}(2) = \varphi_0(3) \leq \varphi_0(4) < t = C_{\hat{A}}(2).$$

Essentially, the actions taken to ensure $C_{\hat{A}}$'s domination in B_0 *took care of* the domination for the first element of B_1. That said, this indicates that the problem that the number 0 posed will not re-appear for other elements when the procedure is generalized to consider all of ω.

As a last example, the case where two partial computable functions, φ_0 and φ_1, and the numbers 0 to 7 are discussed. As before we still have the two blocks B_0 and B_1 but now B_1 consists of the numbers 2 through 7. We show that it is possible to preserve $C_{\hat{A}}$'s domination over $\varphi_0 \circ C_{\tilde{A}}$ in both blocks, as it was in the previous example, while also granting $C_{\hat{A}}$ domination over $\varphi_1 \circ C_{\tilde{A}}$ in the second block. Before, proceeding note that Figure 3.3 on page 21 gives the configurations of $\tilde{A}$ and $\hat{A}$ as they appear during the following procedure.

Since we are considering two partial computable functions B_1 must now be divided into sub-blocks $B_{1,0}$ and $B_{1,1}$ each consisting of three numbers. Indeed, with two functions in play we can no longer wait for φ_0 to converge before proceeding on to the block B_1 since φ_0 may not be total and as such, we potentially will never act for the sake of φ_1. That all said, we now have to perform the enumeration process for $B_{1,1}$ while waiting for φ_0 to converge for the first block. Suppose that the enumeration process for B_0 completes at stage s and

the enumeration process for $B_{1,1}$ completes at stage t. If $\varphi_0(s)$ converges before $\varphi_1(t)$ then the number 0 is removed from both sets which renders the enumeration process performed for $B_{1,1}$ obsolete, as the configurations seen during the enumeration process had both 0 and 1 enumerated in. We now have to perform the enumeration process for the sub-block $B_{1,0}$ to account for the change in B_0.

Further notice that the sub-blocks of B_1 must also consist of three elements rather than two. If each sub-block consisted of two elements and the two functions φ_0 and φ_1 are total then it is possible for the initial segment up to 3 in $\tilde{A}$ and $\hat{A}$ to settle on the configuration appearing at stage t in Figure 3.2 which gives the numbers 4 and 5 the same computation values in $\tilde{A}$ and $\hat{A}$. That is, since $B_{1,0}$ is completely removed from both sets, the actions taken upon it after the enumeration process was performed on $B_{1,1}$ do not affect the computation times of $B_{1,1}$. Now as 4 and 5 have the same computation times, these numbers are problematic as we will want $C_{\hat{A}}(x) > \varphi_0 \circ C_{\tilde{A}}(x)$ for all $x > 0$. To remedy this, the size of each sub-block is increased by one so we do not have the case where an entire sub-block is removed due to the convergence of the functions and as such the actions taken upon $B_{1,0}$ do actually affect the computation times of the elements of $B_{1,1}$. So then $B_{1,0}$ consists of the numbers 2, 3, and 4 and $B_{1,1}$ consists of 5, 6, and 7. We now proceed to build $\tilde{A}$ and $\hat{A}$.

At stage 7, perform the usual enumeration process for B_0 in $\tilde{A}$ and $\hat{A}$ which ends at stage 8. As in the second example, go on to perform the enumeration process for $B_{1,1}$ which completes at stage 11 while waiting for $\varphi_0(8)$ to converge. It is at this stage that the procedure differs from the previous example. As a second partial computable function, φ_1, is now considered when dealing with B_1 we must wait to see which is the first to converge, $\varphi_0(8)$ or $\varphi_1(11)$ and act accordingly. Note that we are not waiting for the convergence $\varphi_0(11)$ since our conventions give that $\varphi_0(8)$ must converge first. Suppose that $\varphi_1(11)$ is the first to converge at some stage t. Now remove the least element of $B_{1,1}$ from both approximations and wait for the convergence of $\varphi_0(8)$. When $\varphi_0(8)$ does converge at some later stage, say stage u, remove 0 from both sets and then proceed to perform the enumeration process for $B_{1,0}$ in $\tilde{A}$ and $\hat{A}$, completing at stage $u + 3$. Now we wait for the convergence of $\varphi_0(u + 3)$ and $\varphi_1(u + 3)$. Suppose that $\varphi_0(u + 3)$ converges first at some stage v, now remove the least element of $B_{1,0}$ from both sets. When $\varphi_1(u + 3)$ later converges at some stage w remove the least remaining element of $B_{1,0}$ from the two sets.

The configurations of $\tilde{A}$ and $\hat{A}$ resulting from the process discussed above appear as

	$\tilde{A}$			$\hat{A}$		
	B_0	$B_{1,0}$	$B_{1,1}$	B_0	$B_{1,0}$	$B_{1,1}$
	01	234	567	01	234	567
Stages $0-6$	00	000	000	00	000	000
Stage 7	01	000	000	10	000	000
Stage 8	11	000	000	11	000	000
Stage 9	11	000	001	11	000	100
Stage 10	11	000	011	11	000	110
Stage 11	11	000	111	11	000	111
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage t	11	000	011	11	000	011
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage u	01	000	011	01	000	011
Stage $u+1$	01	001	011	01	100	011
Stage $u+2$	01	011	011	01	110	011
Stage $u+3$	01	111	011	01	111	011
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage v	01	011	011	01	011	011
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Stage w	01	001	011	01	001	011

Figure 3.3

in Figure 3.3. As can be seen from the diagram, the final configurations of $\tilde{A}$ and $\hat{A}$ at stage w match. Further note that this configuration matches the configuration of $\tilde{A}$ at stage $u+1$ while it is not seen in $\hat{A}$ until stage w. The argument for $C_{\hat{A}}(1) > \varphi_0 \circ C_{\tilde{A}}(1)$ is handled similarly as to before. Now as we are considering two total functions the first two elements of B_1 are removed. Similar to how 0 was a problematic element in B_0 when dealing with φ_0, the two numbers 2 and 3 are problematic for φ_1. However, these elements do not negatively affect $C_{\hat{A}}$'s dominance over $\varphi_0 \circ C_{\tilde{A}}$. Indeed, the computation values of these elements in $\tilde{A}$ and $\hat{A}$ are completely determined by the computation values of the number 1 in $\tilde{A}$ and $\hat{A}$ respectively. So for $1 \leq x \leq 3$ we have

$$\varphi_0 \circ C_{\tilde{A}}(x) = \varphi_0(7) \leq \varphi_0(8) < u = C_{\hat{A}}(x).$$

Now for any $x > 3$ we have $C_{\tilde{A}}(x) = u+1$ while $C_{\hat{A}}(x) = w$. So then for $x > 3$, $\varphi_0 \circ C_{\tilde{A}}(x) = \varphi_0(u+1) \leq \varphi_0(u+3) < v < w = C_{\hat{A}}(x)$.

When considering φ_1 in B_1 we only have $C_{\hat{A}}(x) > \varphi_1 \circ C_{\tilde{A}}(x)$ for the numbers $x > 3$. As mentioned, for $x > 3$, we have that $C_{\tilde{A}}(x) = u+1$ while $C_{\hat{A}}(x) = w$ so then

$$\varphi_1 \circ C_{\tilde{A}}(x) = \varphi_1(u+1) \leq \varphi_1(u+3) < w = C_{\hat{A}}(x).$$

Although B_0 and the first two elements of B_1 are problematic for $C_{\hat{A}}$'s domination of $\varphi_1 \circ C_{\tilde{A}}$, they are the only such elements similar to the case with φ_0 and the number 0.

Now if we had instead assumed that $\varphi_0(8)$ had converged first rather than $\varphi_1(11)$, the configurations above would have appeared the same except that the number 5 would have remained enumerated in both sets. Similar arguments to those above would have shown that $\varphi_0 \circ C_{\tilde{A}}(x) < C_{\hat{A}}(x)$ for $x > 0$ and $\varphi_1 \circ C_{\tilde{A}}(x) < C_{\hat{A}}(x)$ for $x > 3$.

If φ_1 had not converged at all then it would have been as in the second example. If φ_0 had not converged at all then $B_{1,0}$ would not have been acted upon and $\varphi_1 \circ C_{\tilde{A}}(x) < C_{\hat{A}}(x)$ would only hold for $x > 4$. However, this situation is quite like the second example and similar arguments can be made to show that if more blocks were introduced, we would have $\varphi_1 \circ C_{\tilde{A}}(x) < C_{\hat{A}}(x)$ for all elements x of those blocks. Now if neither function converged, only the elements of B_0 and $B_{1,1}$ would have been enumerated into both sets and would not have been removed and as such $\tilde{A}$ and $\hat{A}$ would be equal. Further, since neither function is total they would not be considered when discussing computation time domination. We now

move on to the procedure that actually builds the sets $\tilde{A}$ and $\hat{A}$, and their approximations $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$, to meet the requirements of the theorem.

The more general idea is to divide all of ω into blocks $B_0, B_1, B_2, \ldots$ where B_0 consists of a single sub-block of a pair of bits and for $n > 0$, B_n will have $(n \times b(n-1)) + 1$ many sub-blocks, $B_{n,0}, \ldots, B_{n,n\times b(n-1)}$, such that each sub-block has $n+2$ many bits and $b(n-1)$ denotes the number of sub-blocks of B_{n-1}. The enumeration process discussed in the examples is then performed for these blocks. In particular, the enumeration process for $B_{n,k}$ at stage s is performed by enumerating the elements of $B_{n,k}$ one at a time from greatest to least into $\tilde{A}$ and from least to greatest into $\hat{A}$ finishing at stage $s+n+1$. Let s_n denote the most recent stage at which the enumeration process is completed on a sub-block of B_n.

Now if $\varphi_e(s_n) \downarrow$ for some $0 \leq e \leq n$ at some stage s then remove from both sets the least element of B_n that is still enumerated in $\tilde{A}$ and $\hat{A}$ which will be an element of some sub-block $B_{n,k}$. For the blocks following B_n that have already been acted upon, we shift down a sub-block and perform the enumeration process on this sub-block to provide a new configuration that takes B_n's recent change into account. As we enumerated the elements of $B_{n,k}$ from greatest to least into $\tilde{A}$, the initial segment up to $\max\{B_n\}$ in $\tilde{A}$ has a configuration appearing during the enumeration process for $B_{n,k}$ that matches the configuration appearing at stage s when $\varphi_e(s_n)$ converges. On the other hand, the initial segment in $\hat{A}$ does not have such an earlier matching configuration since the elements of $B_{n,k}$ were enumerated in from least to greatest. That is, this configuration does not appear in $\hat{A}$ until the stage when $\varphi_e(s_n)$ converges.

As B_n has $(n \times b(n-1)) + 1$ many sub-blocks there are enough sub-blocks to allow the shift mentioned above. Indeed, there are enough sub-blocks to allow each $\varphi_i(s_k)$ to converge, where $0 \leq i \leq n-1$ and s_k can take on at most $b(n-1)$ many values (one for each sub-block of B_{n-1}). This accounts for each partial computable function that B_{n-1} takes into consideration and all the stages of which the enumeration process was completed on a sub-block of B_{n-1}. Furthermore, since each sub-block of B_n consists of $n+2$ many bits and B_n only considers the partial computable functions φ_e where $e \leq n$, there are enough elements to remove as the φ_e converge.

Before proceeding to the construction some terminology is explained. A sub-block is declared active if the most recent enumeration process performed on its respective block

was performed on *that* sub-block. As such, a block can only have a single active sub-block at any given moment. That is, when $B_{n,k}$ is activated, $B_{n,k+1}$ will be deactivated. We deem a block active if it has an active sub-block. Additionally, since the following process is a step by step procedure where a single step may involve several stages of action and computation we use $stage(t)$ to keep track of the stage appearing at the start of step t and s to keep track of the current stage.

Construction:

Step 0: Set stage(0):=0 and stage(1):=1.

Step t: Set s:=stage(t). If $\varphi_{e,s}(s_n) \downarrow$ (again, where s_n denotes the most recent stage at which the enumeration process was completed on a sub-block of B_n) for some n, and some $e \leq n$, such that B_n has an active sub-block, $B_{n,k}$, then remove from both $\tilde{A}$ and $\hat{A}$ the least element $x \in B_{n,k}$ such that $x \in \tilde{A}_{s-1}$ and $x \in \hat{A}_{s-1}$, and reset $s := s+1$. Now in ascending order of $m > n$ where B_m has an active sub-block, $B_{m,\ell}$, perform the enumeration process for $B_{m,\ell-1}$ and declare the sub-block $B_{m,\ell-1}$ active. Reset $s_m := s + m + 1$ and $s := s_m$.

In ascending order of the m such that $s > \max\{B_{m+1}\}$ and B_m does not have an active sub-block, declare $B_{m,m\times b(m-1)}$ active and perform the enumeration process for that sub-block. Now define $s_m := s + m + 1$ and reset $s := s_m$.

If neither of the situations above occur then reset $s := s + 1$.

Define stage$(t + 1) := s$.

This completes the construction process.

Note that any element that is enumerated into $\tilde{A}$ and $\hat{A}$ during the enumeration process on a sub-block and then later removed due to the convergence of a partial computable function never enters either approximation again. As such, we have that the sets are, in fact, 2-c.e.

For any sub-block that is activated, all of the elements of that sub-block are enumerated into $\tilde{A}$ and $\hat{A}$ during the enumeration process and any number that is removed due to the convergence of a partial computable function is removed simultaneously from both sets. This being the case, we have that $\tilde{A} = \hat{A}$. That is, $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$ will approximate the same 2-c.e. set as desired.

Further note that as we do not act on block B_n until a stage $s > \max\{B_{n+1}\}$, any action taken on this block will not only affect the computation values of its own elements but the elements of the following block as well. The following lemma shows that $\hat{A}$ does, in fact, computation time dominate $\tilde{A}$.

Lemma 3.5. *If φ_e is total then $(\forall n > e)(\forall x \in B_n)[C_{\hat{A}}(x) > \varphi_e \circ C_{\tilde{A}}(x)]$.*

Proof. Let φ_e be total and consider B_n where $n > e$. Let y be the least element of B_n that is in $\tilde{A}$ and $\hat{A}$. Suppose $B_{n,k}$ is the sub-block such that $y \in B_{n,k}$. Now it must be the case that the blocks preceding B_n have already settled prior to the enumeration process being performed on $B_{n,k}$ otherwise $B_{n,k-1}$ would have been activated which implies that an element of $B_{n,k-1}$ is in $\tilde{A}$ and $\hat{A}$, contradicting the assumption that y is the least element of B_n in $\tilde{A}$ and $\hat{A}$.

First consider the set $\tilde{A}$. Since the elements of B_n are enumerated into $\tilde{A}$ in descending order, we have that y is also the last element of $\tilde{A}$ in B_n that is enumerated in $\tilde{A}$. Now, as the elements of $B_{n,k}$ that are removed are done so in ascending order, for each $x \in B_n$ such that $x \geq y$ we have x is not acted upon after its initial enumeration into the two sets and so, $C_{\tilde{A}}(x) = C_{\tilde{A}}(y)$. Additionally, since the preceding blocks have already settled, the enumeration process performed on $B_{n,k}$ provides $\tilde{A}$ with a configuration of the initial segment up to $\max\{B_n\}$ that appears correct. That said, $C_{\tilde{A}}(y) \leq s_n$. Ergo, for all $x \in B_n$ such that $x \geq y$, $C_{\tilde{A}}(x) = C_{\tilde{A}}(y) \leq s_n$ and consequently, $\varphi_e \circ C_{\tilde{A}}(x) = \varphi_e \circ C_{\tilde{A}}(y) \leq \varphi_e(s_n)$.

Now consider $\hat{A}$. Since φ_e is total we have that $\varphi_e(s_n) \downarrow$ at some stage s. At this stage an element $z < y$, $z \in B_{n,k}$, is removed from $\hat{A}$. Since the elements of $B_{n,k}$ were enumerated into $\hat{A}$ in ascending order, we have that such a configuration has not appeared earlier and as a result, $C_{\hat{A}}(x) \geq s > \varphi_e(s_n)$ for all $x \geq y$ where $x \in B_n$. Hence, for all $x \in B_n$ such that $x \geq y$,

$$C_{\hat{A}}(x) \geq s > \varphi_e(s_n) \geq \varphi_e \circ C_{\tilde{A}}(x)$$

where $\varphi_e(s_n) \geq \varphi_e \circ C_{\tilde{A}}(x)$ holds from the argument in the previous paragraph.

Lastly, for all $x \in B_n$ such that $x < y$, we must have that $x \notin \tilde{A}$ and $x \notin \hat{A}$ since y is the least element of B_n that is in both sets. Since B_{n-1} is not activated until a stage $t > \max\{B_n\}$, the actions taken on B_{n-1} affect the computation values of B_n. Now as $x \notin \tilde{A}$ and $x \notin \hat{A}$ for all $x \in B_n$ such that $x < y$, we have that the computation times of x

are completely determined by B_{n-1}. In particular, we have that $C_{\tilde{A}}(x) = C_{\tilde{A}}(\max\{B_{n-1}\})$ and $C_{\hat{A}}(x) = C_{\hat{A}}(\max\{B_{n-1}\})$. An argument as given above will show that

$$C_{\hat{A}}(\max\{B_{n-1}\}) > \varphi_e \circ C_{\tilde{A}}(\max\{B_{n-1}\}).$$

Thus, for each $x \in B_n$ such that $x < y$, we have $C_{\hat{A}}(x) > \varphi_e \circ C_{\tilde{A}}(x)$. Hence, for all $x \in B_n$, $C_{\hat{A}}(x) > \varphi_e \circ C_{\tilde{A}}(x)$.

□

The lemma verifies that for any computable function, f, $(\forall^\infty x)[C_{\tilde{A}}(x) > f \circ C_{\tilde{A}}(x)]$. Hence, the requirements of the theorem have been met.

□

Although Theorem 3.4 provides a counter-example to the well-definedness of computation time domination on 2-c.e. approximations of 2-c.e. sets, an analogous construction cannot be made for the case of 2-c.e. approximations of c.e. sets, as the next proposition shows.

Proposition 3.6. *Given a c.e. set, A, with two 2-c.e. approximations, $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$, there exists a computable function, f, such that for all x, $f \circ C_{\tilde{A}}(x) \geq C_{\hat{A}}(x)$. In particular, there is a strictly increasing computable function, f', such that for all x, $f' \circ C_{\tilde{A}}(x) \geq C_{\hat{A}}(x)$.*

Proof. Since A is c.e. there exists an index, e, such that $A = W_e$. Now define f as follows:

$$f(s) = (\mu t \geq s)[\tilde{A}_t \restriction s = W_{e,t} \restriction s = \hat{A}_t \restriction s]$$

Such a t in the above definition of f must exist as $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$ approximate A and $A = W_e$. This function meets the requirements of the theorem.

Verification: Let $n = C_{\tilde{A}}(x)$ for some x, note that $n \geq x$ by the definition of the computation function. We have that $\tilde{A}_n \restriction x = W_e \restriction x$ and since $\{\tilde{A}_s\}_{s\in\omega}$ is a 2-c.e. approximation, it is the case that $\tilde{A}_n \restriction x \subseteq \tilde{A}_m \restriction x$ for all $m \geq n$. By the definition of the function f, $\tilde{A}_{f(n)} \restriction n = W_{e,f(n)} \restriction n = \hat{A}_{f(n)} \restriction n$. Then as $f(n) \geq n$ we get,

$$W_e \restriction x = \tilde{A}_n \restriction x \subseteq \tilde{A}_{f(n)} \restriction n = W_{e,f(n)} \restriction n.$$

As W_e is c.e. we have that for all t, $W_{e,t} \Vdash x \subseteq W_e \Vdash x$. That said, $W_e \Vdash x \subseteq W_{e,f(n)} \Vdash x$ gives $W_{e,f(n)} \Vdash x = W_e \Vdash x$. So, $\hat{A}_{f(n)} \Vdash x = W_{e,f(n)} \Vdash x = W_e \Vdash x = A \Vdash x$ and as such, $C_{\hat{A}}(x) \leq f(n) = f \circ C_{\tilde{A}}(x)$.

To obtain a strictly increasing computable function, f', with the desired properties, define $f'(0) = f(0)$ and $f'(n) = \max\{f(n), f'(n-1)+1\}$ for $n \geq 1$.

□

Proposition 3.6 gives that the 2-c.e. set of Theorem 3.4 cannot be a c.e. set and as such, we get the following result for free.

Corollary 3.7. *There exists a properly 2-c.e. set, A, with two 2-c.e. approximations, $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\hat{A}_s\}_{s\in\omega}$, such that for any computable function, f, $(\forall^\infty x)[C_{\hat{A}}(x) > f \circ C_{\tilde{A}}(x)]$.*

This gives that computation time domination is not well defined when considering properly n-c.e. sets when $n > 1$. Now from Proposition 3.6, one sees that no two 2-c.e. approximations of a given c.e. set can computation time dominate one another. However, something much stronger can be stated about 2-c.e. approximations of c.e. sets.

Corollary 3.8. *If A and B are c.e. sets with 2-c.e. approximations $\tilde{A}$, $\hat{A}$, $\tilde{B}$, and $\hat{B}$ such that $\hat{A}$ computation time dominates $\hat{B}$ then $\tilde{A}$ computation time dominates $\tilde{B}$ as well. That is, computation time domination is well-defined in regards to 2-c.e. approximations of c.e. sets.*

Proof. The proof follows analogously from the proof of Theorem 2.1 with the computation function now playing the role of the settling function.

□

As computation time domination is not well defined on n-c.e. sets where $n > 1$, there is no ordering on n-c.e. sets with regards to computation time domination that can be introduced and studied unlike the case with settling time domination on c.e. sets and the structure $\mathcal{E}_{st}$.

Chapter 4

Strong Settling Time Domination

The definition of strong settling time domination takes into regard a second computable function so now rather than just a computable function acting on the settling time of an element x as was the case with settling time domination, we also have a computable function acting on x itself. However, like settling time domination, strong settling time domination is well defined with respect to c.e. sets. In fact much more can be said, strong settling time domination is actually well defined on the *wtt*-degrees with respect to c.e. sets.

Definition 4.1. A set B is weak truth table reducible to a set A, denoted $B \leq_{wtt} A$, if there is a Turing reduction $B = \Phi_e^A$ and a computable function h such that the use of the reduction $\varphi_e^A(x)$ is less than or equal to $h(x)$.

Theorem 4.2 (Csima, Shore [3])**.** *Strong settling time domination is well defined on wtt-degrees. That is, if A, B, and C are c.e. sets with respective associated enumerations $\{A_s\}_{s\in\omega}$, $\{B_s\}_{s\in\omega}$, and $\{C_s\}_{s\in\omega}$ such that $A \leq_{wtt} B <_{sst} C$ or $A <_{sst} B \leq_{wtt} C$ then $A <_{sst} C$.*

Proof. First we will need the following lemma.

Lemma 4.3 (Csima, Shore [3])**.** *Suppose that A and B are c.e. sets such that B is infinite and $A \leq_{wtt} B$ with the use of the reduction bounded by a computable function h. Then there is a strictly increasing computable function f' such that for all x, $m_A(x) \leq f' \circ m_B \circ h(x)$.*

Proof. The proof is similar to Lemma 2.2. □

Suppose that $A \leq_{wtt} B <_{sst} C$. Additionally, let f and g be computable functions. Lemma 4.3 gives a strictly increasing computable function f' such that

$$m_A \circ g(x) \leq f' \circ m_B \circ h \circ g(x),$$

where h is a computable function bounding the use of the *wtt*-reduction from B to A. By our conventions the function f is non-decreasing and as such,

$$f \circ m_A \circ g(x) \leq f \circ f' \circ m_B \circ h \circ g(x).$$

Since h and g are computable so is $h \circ g$. Similarly, $f \circ f'$ is also computable. So then the assumption that $B <_{sst} C$ gives that for almost every x,

$$f \circ m_A \circ g(x) \leq f \circ f' \circ m_B \circ h \circ g(x) < m_C(x).$$

Thus, $A <_{sst} C$.

Now consider the case such that $A <_{sst} B \leq_{wtt} C$. Since $B \leq_{wtt} C$, Lemma 4.3 gives a strictly increasing computable function g' such that for all x,

$$m_B(x) \leq g' \circ m_C \circ h(x),$$

where h is a computable function bounding the use of the *wtt*-reduction from C to B. As g and h are computable so is $g \circ h(x+1)$. Similarly, $g' \circ f$ is also computable. So then the assumption that $A <_{sst} B$ gives that

$$(\forall^\infty x)[g' \circ f \circ m_A \circ g \circ h(x+1) < m_B(x) \leq g' \circ m_C \circ h(x)].$$

Now, from the monotonicity of g' we have

$$(\forall^\infty x)[f \circ m_A \circ g \circ h(x+1) < m_C \circ h(x)].$$

As mentioned earlier, since h is a bound on the use function, we can assume it is non-decreasing. Additionally, we may assume that these sets are not finite and as such, h is not bounded. So then for all y, there is some x such that $h(x) \leq y \leq h(x+1)$. Hence, for all but finitely many y,

$$f \circ m_A \circ g(y) \leq f \circ m_A \circ g \circ h(x+1) \leq m_C \circ h(x) \leq m_C(y).$$

Thus, $A <_{sst} C$ and so strong settling time domination is well defined on the *wtt*-degrees.

□

Corollary 4.4 (Csima, Shore [3])**.** *The strong settling time domination ordering is well defined with regards to c.e. sets.*

Proof. To see that $<_{sst}$ is well defined with regards to c.e. sets, suppose that A and B are c.e. sets such that $A >_{sst} B$. Further, suppose that $\{\tilde{A}_s\}_{s\in\omega}$ and $\{\tilde{B}_s\}_{s\in\omega}$ are approximations respectively for A and B. Since $\{\tilde{A}_s\}_{s\in\omega}$ approximates A we have that $A \equiv_{wtt} \tilde{A}$ with the use of the reduction bounded by the identity function. Similarly, $\tilde{B} \equiv_{wtt} B$. Theorem 4.2 then gives that $\tilde{B} <_{sst} \tilde{A}$.

□

With strong settling time domination being well defined with regards to c.e. sets, we now proceed to study the structure $\mathcal{E}_{sst}$ of c.e. sets with the relation $<_{sst}$. First, we show that $\mathcal{E}_{sst}$ is non-trivial.

Theorem 4.5 (Csima, Shore [3])**.** *There exist c.e. sets A and B such that $A >_{sst} B$ and B is non-computable.*

Proof. A and B are built stage by stage meeting the following:

$R_{\langle i,j\rangle}$: If φ_i and ψ_j are total then for all but finitely many x, $m_A(x) > \varphi_i \circ m_B \circ \psi_j(x)$.

N_e: $B \neq \Phi_e$.

Note that φ_i and ψ_j respectively play the roles of the functions f and g in the definition of strong settling time domination and the reason for the difference in notation is due to this difference in their roles. This notational difference is solely for the sake of clarity. Indeed, we use the same enumeration of the partial computable functions for φ_i and ψ_j.

To ensure that the $R_{\langle i,j\rangle}$ requirements are met we will use markers, Γ_l, with guards, $[i,j,k]_l$, between Γ_{l-1} and Γ_l such that $0 \leq i,j,k \leq l$. The Γ_l^p guard $[i,j,k]_l$ (where $k > 0$) is the guard that corresponds to the functions φ_i and ψ_j and the k^{th} possible entrant of B less than s where s denotes the stage when $\psi(\Gamma_{l+1})\downarrow$. The guard Γ_l^p $[i,j,0]_l$ is the guard

that corresponds to the functions φ_i and ψ_j and will enter A even if there are no elements entering B that are less than s.

Now if $\psi_j(\Gamma_{l+1}) \downarrow$ at stage s then all of the elements of lower priority than Γ_l appointed to enter B will be moved to be greater than s. The commitment is then made that whenever an element less than or equal to s enters B at a stage $t > s$ there will be a Γ_l guard that will enter A at the stage when $\varphi_i(t) \downarrow$. Further, an element is enumerated into A at the stage when $\varphi_i(s) \downarrow$ as well. Note that all of the markers and guards are distinct from one another. For the sake of the N_e requirements there are numbers $x_0 < x_1 < \ldots$ such that for each total Φ_e we have that $B(x_e) \neq \Phi_e(x_e)$. We now proceed to the construction of A and B.

Construction:

Stage 0: Set $x_m^0 = m$ for all $m \geq 0$ and place the Γ_l markers spread out enough to provide room for their guards.

Stage $s+1$: For the least e such that N_e is unsatisfied and $\Phi_e(x_e^s) \downarrow = 0$, enumerate x_e^s into B and declare N_e satisfied. If $\psi_j(\Gamma_{l+1}) \downarrow$ where $j \leq l$, set all x_e^{s+1} to be greater than $s+1$ for all $e \geq l$ such that N_e is unsatisfied. Make the commitment that $[i, j, 0]_l$ will enter A at the stage when $\varphi_i(s+1) \downarrow$. Let $t_1,\ t_2, \ldots,\ t_n$ denote the stages greater than $s+1$ such that an $x_e \leq s+1$ enters B. Note that since all x_e such that $e \geq l$ and N_e is unsatisfied have been moved, we have that $n \leq l$. Declare that at the stage when $\varphi_i(t_k) \downarrow$, $[i, j, k]_l$ is enumerated into A.

Lemma 4.6. *The N_e are satisfied.*

Proof. It is the case that x_e^s is moved only when $\psi_j(\Gamma_{l+1}) \downarrow$ for the first time for some partial computable function ψ_j and marker Γ_{l+1} such that $j \leq l \leq e$. As that may only occur a finite number of times we have that $\lim_{s \to \infty} x_e^s < \infty$. Since all the x_e eventually settle we have that if Φ_e is total then $\Phi_e(x_e) \neq B(x_e)$. Indeed, if $\Phi_e(x_e) = 0$ then x_e is enumerated into B and x_e does not enter B otherwise.

□

Lemma 4.7. *The $R_{\langle i,j \rangle}$ are satisfied.*

Proof. Suppose φ_i and ψ_j are total. It suffices to show that for all but finitely many x that whenever some $y \leq \psi_j(x)$ enters B at some stage s then some number less than or

equal to x will be enumerated into A at some later stage $t > \varphi_i(s)$. Let x be such that $\Gamma_l \leq x \leq \Gamma_{l+1}$ for some $l \geq i, j$. From the construction process it is declared that at the stage s when $\psi_j(\Gamma_{l+1}) \downarrow$ there are Γ_l guards assigned with the commitment of entering A at the stage when $\varphi_i(t) \downarrow$, where t denotes a stage greater than s such that an entrant less than or equal to $\psi_j(\Gamma_{l+1})$ enters B. Further, there are guards assigned to enter A even if there are no future entrants enumerated into B that are less than $\psi_j(\Gamma_{l+1})$.

□

The previous lemmas give that the c.e. sets A and B are as desired.

□

In addition to showing that $\mathcal{E}_{sst}$ is non-trivial, Theorem 4.5 demonstrates the technique used to build c.e. sets such that one strongly settling time dominates the other. Drawing from this technique, it has been shown that some linear orderings may be embedded into $\mathcal{E}_{sst}$.

Theorem 4.8 (Csima, Shore [3])**.** *Given a computable partial ordering P on $\mathbb{N}$ with no infinite ascending sequence there is a computable sequence $\{A_p\}_{p\in\omega}$ of c.e. sets such that if $q <_P p$ then $A_q <_{sst} A_p$.*

In fact, we show here it is also possible to embed any finite partial ordering into $\mathcal{E}_{sst}$. Notice that as these are partial orders rather than linear orders we must preserve the incomparability amongst the elements as well. Indeed, Theorem 4.8 is only concerned with the preservation of the comparability amongst the elements.

Theorem 4.9. *Every finite partial order can be embedded into $\mathcal{E}_{sst}$.*

Proof. Let P be a finite partial ordering with M-many elements. We may assume without loss of generality that the elements of P are $1,\ 2, \ldots,\ M$. We shall build c.e. sets A_1, A_2,..., A_M stage by stage meeting the following requirements:

$R_{p>_P q}$: If $p >_P q$ then for all computable functions f and g we have $(\forall^\infty x)[m_{A_p}(x) > f \circ m_{A_q} \circ g(x)]$. That is, if $p >_P q$ then $A_p >_{sst} A_q$.

$R_{p\,|_P\,q}$: If $p\,|_P\,q$ then $(\exists^\infty x)[m_{A_q}(x) \geq m_{A_p}(x)]$ and $(\exists^\infty x)[m_{A_p}(x) \geq m_{A_q}(x)]$. That is, if $p\,|_P\,q$ then $A_p\,|_{sst}\,A_q$.

To meet the $R_{p>_P q}$ requirements we use a technique similar to that used in Theorem 4.5 to build c.e. sets such that one strong settling time dominates the other. That is, for the sake of the requirement $R_{p>_P q}$, we use domination markers Γ_l^p, $l \in \omega$, with guards $[i,j,c]_l^p$ between Γ_{l-1}^p and Γ_l^p. Similar to Theorem 4.5, the marker Γ_l^p requires a guard corresponding to φ_i and ψ_j such that $i,j \leq l+1$ and each possible entrant less than $\psi_j(\Gamma_{l+1}^p)$ of an A_q such that $p >_P q$. Additionally, Γ_l^p also requires a guard corresponding to φ_i and ψ_j each time Γ_{l+1}^p moves. So then the Γ_l^p guard $[i,j,c]_l^p$ corresponds to φ_i and ψ_j and c is some number less than the sum of the number of times Γ_{l+1}^p moves and the number of possible entrants less than $\psi_j(\Gamma_{l+1}^p)$ of the A_q such that $p >_P q$.

Now if $\psi_j(\Gamma_{l+1}^p)\downarrow$ at a stage s for some partial computable function ψ_j such that $j \leq l+1$ then all of the possible entrants of A_q with lower priority than Γ_{l+1}^p (an explanation on the priority of the entrants is given on page 34) will be moved to be greater than s. Note that when a marker is moved to be greater than s, all of its guards are also moved to be greater than s (in general, when it is said that a marker is greater than another marker it is meant that all of its guards will be greater than that marker as well). The commitment is then made that whenever an element less than or equal to s enters A_q at a stage $t > s$, there will be some Γ_l^p guard, $[i,j,c]_l^p$, that will enter A_p at the stage when $\varphi_i(t)\downarrow$ where $i \leq l+1$. In addition, a guard is enumerated into A_p at the stage when $\varphi_i(s)\downarrow$ even if there are no elements less than s that are later enumerated into A_q.

Now if ψ_j and φ_i are total and x is such that $\Gamma_l^p \leq x \leq \Gamma_{l+1}^p$ for some l where $i,j \leq l+1$ then by our conventions it is the case that $\psi_j(x) \leq \psi_j(\Gamma_{l+1}^p)$. So from the process above, if $\psi_j(\Gamma_{l+1}^p)$ converges at stage s then whenever a number less than or equal to $\psi_j(x)$ (and hence less than s) enters A_q at a stage $t > s$, a Γ_l^p guard (which is less than x) enters A_p at the stage when $\varphi_i(t)\downarrow$ and a guard also enters A_p at the stage when $\varphi_i(s)\downarrow$, even if no elements less than $\psi_j(\Gamma_{l+1}^p)$ enter A_q after stage s. So then we have that $m_{A_p}(x) > \varphi_i \circ m_{A_q} \circ \psi_j(x)$.

To meet the $R_{p\,|_P\,q}$ requirements we use techniques similar to those used in Example 2.4 to preserve any incomparability among elements. That is, if $p\ |_P\ q$ then we use incomparability markers n_k^q and n_k^p, $k \in \omega$, and their associated guards to ensure that $A_p\ |_{sst}\ A_q$. In particular, the guards of n_k^q are less than n_k^q and have the commitment that whenever an element less than or equal to n_k^q enters A_p after n_k^q has been enumerated into A_q, a guard is enumerated into A_q. These actions ensure that there are infinitely many x, namely the n_k^q, such that $m_{A_q}(x) \geq m_{A_p}(x)$ and as such $A_p \not>_{sst} A_q$. Similar commitments are made for the guards of n_k^p. These actions ensure that $A_q \not>_{sst} A_p$ and as a result, we

have that $A_p \mid_{sst} A_q$.

Unlike in Example 2.4 where we enumerated the incomparability marker n_k^q into A_q at stage s for all $q \in P$ such that there is some $p \in P$ where $p \mid_P q$, we instead, at a given stage, only enumerate the marker n_k^q for a single fixed q such that there is some element in P incomparable to q. In a sense, the incomparability markers associated to the sets A_1, $A_2, \ldots,$ and A_M take turns based on the increasing order of the indices of their associated c.e. set with respect to the usual ordering on the naturals. Indeed, at each stage s we act for the n_k^q such that $s = Mk + q$ (recall that P has M-many elements) and there is some $p \in P$ where $p \mid_P q$.

Notice that meeting these requirements suffices when embedding a finite partial ordering into $\mathcal{E}_{sst}$. Indeed, we have that if $p >_P q$ then $A_p >_{sst} A_q$ from the requirement $R_{p>_P q}$. Note that if $A_p >_{sst} A_q$ then we have that $A_q \not>_{sst} A_p$ as well since $>_{sst}$ is a strict ordering. Furthermore, if $p \not>_P q$ then there are two possible cases, one where $q >_P p$ and the other where $p \mid_P q$. In the first case, we meet the requirement $R_{q>_P p}$, giving that $A_q >_{sst} A_p$ and as a result $A_p \not>_{sst} A_q$. Now in the second case the requirement $R_{p \mid_P q}$ is met and again we get $A_p \not>_{sst} A_q$. Thus, we have $p >_P q$ if and only if $A_p >_{sst} A_q$ as desired.

A brief outline of the interaction and priority between the possible entrants stated above is now given. Note that if a marker is moved, its guards will also move along with it. Consider the marker Γ_l^p. If $\psi_j(\Gamma_l^p)$ converges at stage s for some $j \leq l$ then all $\Gamma_{l'}^q$ such that $p >_P q$ and $l \leq l'$ will be moved to be greater than s. Additionally, all n_k^q such that $p >_P q$ and $l < Mk + q$ will be moved to be greater than s. These movements are done so as to keep the number of possible entrants of A_q that are less than $\psi_j(\Gamma_l^p)$ bounded and in doing so, keeping the number of guards that Γ_{l-1}^p will require bounded. That all said, if $\psi_j(\Gamma_l^p)$ converges the markers in A_q such that $p >_P q$ that will not be moved are all $\Gamma_{l'}^q$ such that $l' < l$ and all n_k^q such that $Mk + q \leq l$ and as such Γ_{l-1}^p will require guards for all of these markers and their associated guards.

Now if the marker n_k^p is moved then so will all Γ_l^q such that $p \mid_P q$ and $Mk + p \leq l$. In addition, all incomparability markers $n_{k'}^q$ such that:

1. $k < k'$ in the case that $p \mid_P q$ and $q < p$ in the ordering of the naturals or
2. $k \leq k'$ in the case that $p \mid_P q$ and $p < q$ in the ordering of the naturals

will be moved. Similar to the case with the domination markers, these movements are done to keep the number of possible entrants of A_q that are less than n^p_k bounded and as such, keep the number of guards of n^p_k bounded. Now the only entrants of the set A_q, where $p \mid_P q$, that are less than n^p_k will be the markers $n^q_{k'}$ and Γ^q_l such that $l < Mk + p$ and $k' \leq k$ in the case that $q < p$ or $k' < k$ in the case that $p < q$. Thus, n^p_k requires guards for these markers and their associated guards. Notice that the incomparability marker n^p_k can only move due to the convergence of a partial computable function on a domination marker of higher priority as outlined in the paragraph above. That said, the movement of any marker in this procedure is, ultimately, due to the convergence of a partial computable function on a domination marker.

Note that the order amongst the markers being moved is maintained. That is, if $\Gamma^q_{l'}$ is moved then it will be maintained that $\Gamma^q_{l'} < \Gamma^q_{l'+1}$. Additionally, if $q \mid_P r$ and $r < q$ then if n^r_k and n^q_k are both moved then the markers will be moved so that $n^r_k < n^q_k$ continues to hold. Further, it will be maintained that $n^q_k < n^q_{k+1}$. Lastly, it will be maintained that $n^p_k < \Gamma^q_l$ if $p \mid_P q$ and $Mk + p \leq l$. Again, note that when a marker is greater than another marker it is implied that all of its guards are greater than the marker as well. These relationships must be maintained so that we may compute exactly which possible entrants are less than a given marker. Indeed, by maintaining these relationships, the possible entrants that a marker requires guards to account for will precisely be the entrants of higher priority.

With the priority of the possible entrants decided we may now begin the process of defining the computable functions $h(p, l)$ and $g(p, k)$ that, respectively, bound the number of times that the domination markers, Γ^p_l, and incomparability markers, n^p_k, can move. We first define the computable functions $\eta_h(p, l)$ and $\eta_g(p, k)$ that give the indices of the domination markers that can move Γ^p_l and n^p_k respectively. That is, $\eta_h(p, l)$ gives the indices, (q, k), of the domination markers such that the convergence of a partial computable function ψ_j, $j \leq k$, on Γ^q_k results in Γ^p_l moving. Similarly, $\eta_g(p, k)$ gives the indices of the domination markers that can potentially move the marker n^p_k. With the knowledge of which markers can move each other we show that each time a marker has to move, it only moves a finite distance. From this we have that it is worthwhile to show that each marker only moves finitely many times which ultimately gives that each marker eventually settles.

Recall from the explanation of the interaction of the markers that for a given marker Γ^p_l, the markers $\Gamma^q_{l'}$ such that $q >_P p$ and $l' \leq l$ and the markers n^r_k such that $r \mid_P p$ and $Mk + r \leq l$ can move it. Further recall that the incomparability markers do not move of

their own accord. That is, a marker, n^r_k, ultimately only moves due to the convergence of some partial computable function on some domination marker. The function $\eta_h(p,l)$ gives the indices of the domination markers that have the potential to move Γ^p_l whether it be directly or indirectly.

Now consider the incomparability marker n^q_k. From our priority, n^q_k can be directly moved by markers of the form Γ^p_l where $p >_P q$ and $l < Mk+q$ and the markers $n^r_{k'}$ where $r \mid_P q$ and $k' < k$ if $q < r$ or $k' \leq k$ if $r < q$. Again, the incomparability markers ultimately only move due to the convergence of a partial computable function on a domination marker. That all said, $\eta_g(q,k)$ is the collection of the indices of the domination markers that can directly or indirectly move n^q_k.

Notice that as Γ^p_l can be moved by the incomparability markers n^q_k such that $q \mid_P p$ and $Mk+q \leq l$, the function η_h will be recursive in η_g since it must take into account all the possible domination markers that move such n^q_k. Likewise, since the incomparability markers can move each other, the function η_g will be recursive in itself.

Define the functions η_h and η_g as:

$$\eta_h(p,l) = \{(q,l') : (p <_P q) \wedge (l' \leq l)\} \cup \bigcup_{\substack{r \mid_P p \\ and\ Mk+r \leq l}} \eta_g(r,k)$$

and

$$\eta_g(p,k) = \{(q,l) : (l < Mk+p) \wedge (p <_P q)\} \cup \bigcup_{\substack{r \mid_P p \\ and\ k' < k}} \eta_g(r,k') \cup \bigcup_{\substack{r \mid_P p \\ and\ r < p}} \eta_g(r,k)$$

An explanation of the definitions of the functions is now given and then we will proceed to show that the sets of indices given by these functions are computable and finite. First consider the definition of the function $\eta_h(p,l)$. The first set in the union gives the indices of the domination markers that can directly move the marker Γ^p_l. That is, it gives the indices (q,l') of all the domination markers, $\Gamma^q_{l'}$ such that $q >_P p$ and $l' \leq l$. Recall that the convergence of a partial computable function ψ_j on $\Gamma^q_{l'}$ such that $q >_P p$, $j \leq l'$, and $l' \leq l$ results in Γ^p_l moving. The union

$$\bigcup_{\substack{r \mid_P p \\ and\ Mk+r \leq l}} \eta_g(r,k)$$

in the definition of $\eta_h(p,l)$ collects the indices of the domination markers that can move the incomparability markers n_k^r that can move Γ_l^p. Recall that the markers of the form n_k^r such that $r \mid_P p$ and $Mk + r \leq l$ are of higher priority than Γ_l^p and as such, can move Γ_l^p. As we will show, $\eta_g(r,k)$ gives the indices of the domination markers that can move the marker n_k^r and thus, gives the indices of the domination markers that can indirectly move Γ_l^p by moving n_k^r.

Essentially, $\eta_h(p,l)$ gives the indices of the domination markers that can directly move Γ_l^p and the indices of the domination markers that can move the incomparability markers that can directly move Γ_l^p. Now if $\Gamma_{l'}^q$ moves Γ_l^p it can only do so directly or by ultimately moving an incomparability marker that directly moves Γ_l^p and as such $\eta_h(p,l)$ will give the indices of all the domination markers that can move Γ_l^p.

Now consider the definition of the function $\eta_g(p,k)$. The first set in the union gives the indices of the domination markers that can directly move the incomparability marker n_k^p. Indeed, the convergence of a partial computable function ψ_j on a domination marker, Γ_l^q such that $j \leq l < Mk + p$ and $q >_P p$ results in n_k^p moving. Now notice that if $r \mid_P p$ and $k' < k$ then when $n_{k'}^r$ moves so will n_k^p. The union

$$\bigcup_{\substack{r \mid_P p \\ and\ k' < k}} \eta_g(r,k')$$

gives the indices of the domination markers that can move these incomparability markers. That is, it gives the indices of the domination markers that can indirectly move n_k^p through moving the previously mentioned $n_{k'}^r$. Now in the case that $r < p$ in the usual ordering and $r \mid_P p$ the marker n_k^r can move n_k^p. That said, the union

$$\bigcup_{\substack{r \mid_P p \\ and\ r < p}} \eta_g(r,k)$$

gives the indices of the domination markers that can indirectly move n_k^p through such n_k^r. So then $\eta_g(p,k)$ gives the indices of the domination markers that can either directly move n_k^p or can ultimately move an incomparability marker that can directly move n_k^p. We now inductively show that $\eta_h(p,l)$ and $\eta_g(p,k)$ are computable and give finite sets.

Lemma 4.10. *The functions $\eta_h(p,l)$ and $\eta_g(p,k)$ are computable and give finite sets.*

Proof. Consider the marker Γ_0^p. From our priority we have that Γ_0^p can only be moved by the convergence of the partial computable function ψ_0 on a marker Γ_0^q such that $q >_P p$. Now as the partial order P is finite, the set $\{(q,0) : p <_P q\}$ will be finite and as such $\eta_h(p,0)$ is finite. Moving on, consider the incomparability marker n_0^1. It is the case that n_0^1 only moves due to the convergence of ψ_0 on some Γ_0^p such that $p >_P 1$ and as such,

$$\eta_g(1,0) = \{(p,0) : p >_P 1\}.$$

Thus, we have that $\eta_g(1,0)$ is computable and gives a finite set.

Continuing inductively, consider $\eta_h(p,l)$ and $\eta_g(q,k)$ such that $Mk+q = l+1$. Suppose that $\eta_h(p',l')$ and $\eta_g(r,k')$ have been computed and are finite for $p' \in P$, $l' < l$, and $Mk' + r \leq l$. The marker Γ_l^p can only directly be moved by markers of the form $\Gamma_{l'}^q$ and $n_{k'}^r$ where $q >_P p$, $p \mid_P r$, $l' \leq l$, and $Mk' + r \leq l$. We have that the set of indices of these $\Gamma_{l'}^q$ is finite as $q \in P$ and $l' \leq l$. From our induction hypothesis $\eta_g(r,k')$ is computable and finite and as such, the union

$$\bigcup_{\substack{r \mid_P p \\ and\ Mk'+r \leq l}} \eta_g(r,k')$$

is finite and computable. Thus, $\eta_h(p,l)$ gives the finite set of all the indices of the domination markers that can move Γ_l^p.

Moving on, consider $\eta_g(q,k)$. The marker n_k^q can be directly moved by markers of the following forms:

1. $\Gamma_{l'}^p$ such that $p >_P q$ and $l' < Mk + q$
2. $n_{k'}^r$ such that $r \mid_P q$ and $k' < k$
3. n_k^r such that $r \mid_P q$ and $r < q$.

The set of indices of the domination markers of the form $\Gamma_{l'}^p$, that can directly move n_k^q, is finite since P is finite and $l' < Mk + q$. From our induction hypothesis, $\eta_g(r,k')$ computes a finite set since $Mk' + r < Mk + q = l + 1$. Along the same line, we have that $\eta_g(r,k)$ computes a finite set since $Mk + r < Mk + q = l + 1$. So then the unions,

$$\bigcup_{\substack{r \mid_P p \\ and\ k'<k}} \eta_g(r,k') \text{ and } \bigcup_{\substack{r \mid_P q \\ and\ r<q}} \eta_g(r,k)$$

are finite. Now any domination marker that can move n^q_k does so either directly or indirectly by ultimately moving a marker of the form (2) or (3) from above. That being the case, if a domination marker can move n^q_k, its index will be in the set

$$\{(p,l) : (l < Mk + q) \wedge (q <_P p)\} \cup \bigcup_{\substack{r \mid_P q \\ \text{and } k' < k}} \eta_g(r,k') \cup \bigcup_{\substack{r \mid_P q \\ \text{and } r < q}} \eta_g(r,k).$$

In particular, the index of such a marker will be in the finite set given by $\eta_g(q,k)$. Thus, we have shown that the functions $\eta_h(p,l)$ and $\eta_g(q,k)$ are computable and give finite sets consisting of the indices of the domination markers that can respectively move the markers Γ^p_l and n^q_k. □

Now that we have determined that a given marker can be moved by only finitely many domination markers we now proceed to show that each time a marker does move, it only moves finitely much. First consider the markers of the form Γ^p_0. Now if ψ_0 converges at stage s on some Γ^q_0 such that $q >_P p$ then Γ^p_0 will be moved such that its least guard will be greater than s. Recall from the explanation on the priority and ordering of the markers on page 34 that Γ^p_0 only has to be greater than s. That is, there are not any markers that have to be less than Γ^p_0 aside from Γ^q_0 which will not have to move and is less than s. That said, whenever any marker of the form Γ^p_0 moves, it only moves a finite amount.

Moving on, consider the incomparability marker n^1_0. Similar to the case with the marker Γ^p_0, if n^1_0 is forced to move due to the convergence of the partial computable function ψ_0 on some marker Γ^q_0 such that $q >_P 1$ at stage s, it will only have to move so that its least guard will be greater than s. Again the priority and ordering of the markers does not require any markers to be less than n^1_0 when it is moved other than Γ^q_0, which will not have to move and will be less than s.

Continuing inductively, consider the markers Γ^p_l and n^q_k such that $Mk + q = l + 1$. Suppose that for all $\Gamma^p_{l'}$ where $p \in P$ and $l' < l$ that if a partial computable function ψ_j converges on some Γ^r_m such that $(r,m) \in \eta_h(p,l')$ and $j \leq m$ then $\Gamma^p_{l'}$ only moves a finite distance. Further, suppose for all $n^r_{k'}$ where $Mk' + r \leq l$ that if a partial computable function ψ_j converges on some $\Gamma^{r'}_m$ such that $(r',m) \in \eta_g(r,k')$ and $j \leq m$ then $n^r_{k'}$ only has to move a finite amount.

First we show that whenever the marker Γ^p_l does move, it only moves a finite distance. If Γ^p_l has to move it will be due to the convergence of a partial computable function ψ_j at

some stage s on a $\Gamma^{q'}_{l'}$ such that $j \leq l'$ and $(q', l') \in \eta_h(p, l)$. Now from the ordering and priority of the markers, Γ^p_l moves to be greater than s, any markers Γ^p_m where $m < l$ that are moved, and any markers $n^r_{k'}$ where $r \mid_P p$ and $Mk' + r \leq l$ that are moved. From our induction hypothesis these markers only move a finite distance and so, Γ^p_l only has to move a finite amount. Now if there is a convergence of a partial computable function at stage s on a marker $\Gamma^{p'}_l$ such that $p' >_P p$, Γ^p_l will only have to move to be greater than s since there are not any markers that are moved that have to be less than Γ^p_l. Thus, whenever the marker Γ^p_l has to move, it only moves a finite distance.

Now consider the marker n^q_k. From our priority if the marker $\Gamma^p_{l'}$ has the ability to move n^q_k then $l' < Mk + q = l + 1$. If $l' < l$ and ψ_j converges on $\Gamma^p_{l'}$ such that $j \leq l'$ at stage s then from the priority and ordering of the markers n^q_k will have to be moved to be greater than:

1. The number s
2. Any marker of the form $n^r_{k'}$, where $r \mid_P q$ and $Mk' + r < Mk + q$, that also has to move
3. Any marker of the form $n^q_{k'}$, where $k' < k$, that also has to move.

Notice that n^q_k only has to be moved to be greater than s and a finite number of markers which, from our induction hypothesis, only have to move a finite amount. That is, n^q_k only has to move a finite distance. Now consider the case where ψ_j converges at stage s on a Γ^p_l such that $j \leq l$ and $p >_P q$. In this situation n^q_k only has to be moved to be greater than s as there are no markers that are moved that have to be less than n^q_k. Thus, n^q_k will only ever have to move a finite distance.

Now that we have completely determined that only a finite a number of markers can move a given marker and that any such movement is finite we go on to show that these markers only move a finite number of times. This gives that all of our markers will eventually settle. We finally define the computable functions $h(p, l)$ and $g(p, k)$ that, respectively, bound the number of times that the domination markers, Γ^p_l, and incomparability markers, n^q_s, will move to be:

$$h(p, l) = \sum_{(q,k) \in \eta_h(p,l)} [(h(q, k) + 1)(k + 1)]$$

and

$$g(p,k) = \sum_{(q,l)\in\eta_g(p,k)} [(h(q,l)+1)(l+1)].$$

First consider the function $h(p,l)$. We have that $\eta_h(p,l)$ is the collection of all indices (q,k) such that Γ_k^q has the ability to move the marker Γ_l^p. Indeed, the convergence of some partial computable function ψ_j on such a marker Γ_k^q where $j \leq k$ results in Γ_l^p moving. That said, given such a Γ_k^q, we only consider $(k+1)$-many partial computable functions and Γ_k^q can only move $h(q,k)$-many times. Hence, this convergence can only occur at most $((h(q,k)+1)(k+1))$-many times.

Continuing, consider the function $g(p,k)$. The function $\eta_g(p,k)$ gives the collection of all indices (q,l) such that the convergence of some partial computable function ψ_j on Γ_l^q where $j \leq l$ can move the marker n_k^p. From an argument similar to the one in the previous paragraph we have that this can occur at most $((h(q,l)+1)(l+1))$-many times.

We now show that $h(p,l)$ and $g(p,k)$ are both computable and finite. First recall that if the partial computable function ψ_j converges on the marker Γ_l^p where $j \leq l$ then the markers that will be moved are

1. $\Gamma_{l'}^q$ such that $p >_P q$ and $l \leq l'$
2. n_k^r such that $p >_P r$ and $l < Mk + r$.

Notice that none of the incomparability guards, n_k^r, that move have the priority to move Γ_l^p since $l < Mk + r$. Now as the partial ordering P is a strict partial ordering and $p >_P q$, it cannot be the case that $q >_P p$. That said, the convergence of a partial computable function on the marker Γ_l^q does not result in Γ_l^p moving. So then the convergence of a partial computable function on Γ_l^p does not ultimately result in Γ_l^p moving. We now inductively show that any given marker can move only finitely many times. That is, we show that $h(p,l)$ and $g(p,k)$ are finite.

Lemma 4.11. *The functions $h(p,l)$ and $g(p,k)$ are finite.*

Proof. If p is a maximal element with regards to the ordering $<_P$ then Γ_0^p never moves and as such, $h(p,0) = 0$. With the movements determined for the maximal elements, we

may inductively determine $h(p,0)$ for all $p \in P$ since Γ_0^p can only be moved due to the convergence of the partial computable function, ψ_0, on a Γ_0^q such that $q >_P p$. Then having determined $h(p,0)$ for all $p \in P$ we can compute $g(1,0)$ since the marker n_0^1 can only be moved by the convergence of ψ_0 on a Γ_0^p such that $p >_P 1$.

Continuing inductively, consider $h(p,l)$ and suppose that $h(q,k)$ has been determined for all $q \in P$ and $k < l$. With such $h(q,k)$ determined we can compute $h(p,l)$ for all maximal p since Γ_l^p can only be moved by markers of the form $n_{k'}^q$ such that $q \mid_P p$ and $Mk' + q \leq l$. These $n_{k'}^q$, in turn, can, ultimately, only be moved by the convergence of some partial computable function on a marker $\Gamma_{l'}^r$ such that $l' < Mk' + q$ (and hence $l' < l$) and either $r >_P q$ or there is some p' where $r >_P p'$ and $p' \mid_P q$ (this is the case where $\Gamma_{l'}^r$ has the ability to indirectly move $n_{k'}^q$). Regardless of the situation, any movement of Γ_l^p is, ultimately, due the convergence of a partial computable function on some Γ_k^q such that $k < l$ and by our assumption we have bounded the number of times Γ_k^q can move and so we may determine $h(p,l)$. Again, with the number of movements determined for the maximal elements we can inductively determine $h(p,l)$ for all $p \in P$.

Now consider $g(q,k)$ and suppose we have determined $h(p,l)$ for all $p \in P$ and $l < Mk + q$. From our discussion on the priority of the markers and their interaction, it is the case that n_k^q, essentially, is only moved due to the convergence of a partial computable function on some Γ_l^p such that $l < Mk + q$. From our assumption, we have determined the number of times these markers can move and thus we can, in turn, compute $g(q,k)$.

From the arguments above we can inductively compute $h(p,l)$ and $g(q,k)$ for all $p, q \in P$ and $k, l \in \omega$. Thus, any given marker can only move finitely many times and as such, will eventually settle. □

Since we have decided the priority of the markers and have defined functions that bound the number of times they may move, we can now define the computable functions counting the number of guards the markers will require. If we can determine the number of guards that a marker requires we will be able to assign the correct number of guards to our markers at the start of our procedure and place the markers such that there will be room for their guards. Additionally, we must guarantee that any given marker will only require a finite number of guards to ensure that this process will be successful. The function $\alpha(p,l)$ counts the number of guards needed for the domination markers, Γ_l^p, while the function $\beta(p,k)$ counts the number of guards needed for the incomparability markers, n_k^p. First declare

that $\alpha(p,l) = 0$ if there is no element $q \in P$ such that $p >_P q$. Additionally, we have that $\beta(p,k) = 0$ if there is no $q \in P$ such that $p \mid_P q$.

$$\alpha(p,l) = \sum_{\substack{Mk+q \leq l+1, \\ \text{and } q <_P p}} [(\beta(q,k)+1)(l+2)^2(g(q,k)+1)] + \sum_{\substack{k \leq l, \\ \text{and } q <_P p}} [(\alpha(q,k)+1)(l+2)^2(h(q,k)+1)]$$
$$+ (h(p,l+1)+1)(l+2)^2$$

and

$$\beta(q,k) = \sum_{\substack{l < Mk+q, \\ \text{and } p \mid_P q}} [\alpha(p,l)(h(p,l)+1)] + \sum_{\substack{Mk'+p < Mk+q, \\ \text{and } p \mid_P q}} \beta(p,k')$$

First consider the definition of the function $\alpha(p,l)$. Following the plan outlined earlier, if $q <_P p$ and $\psi_j(\Gamma^p_{l+1}) \downarrow$ at some stage s for some ψ_j such that $j \leq l+1$ then when $\varphi_i(t) \downarrow$ where $i \leq l+1$ and t denotes a stage where an element less than or equal to s enters A_q, a Γ^p_l guard is enumerated into A_p. The first sum in the definition of $\alpha(p,l)$ counts the number of guards needed to account for the possible entrants of A_q (such that $p >_P q$) due to A_q's incomparability markers and their associated guards of higher priority than Γ^p_{l+1}. Indeed, $(\beta(q,k)+1)$ accounts for such an n^q_k and its guards, $(l+2)^2$ accounts for the convergence of the partial computable functions φ_i and ψ_j such that $i,j \leq l+1$, and $(g(q,k)+1)$ accounts for the number of times n^q_k may move. Similarly, the second sum in the definition of $\alpha(p,l)$ counts the number of guards needed to account for the possible entrants of A_q due to A_q's domination markers of higher priority than Γ^p_{l+1} and their associated guards. Lastly, the expression $(h(p,l+1)+1)(l+2d)^2$ counts the number of guards needed to guarantee domination even if A_q has no possible entrants less than $\psi_j(\Gamma^p_{l+1})$ where $j \leq l+1$. That is, if $\psi_j(\Gamma^p_{l+1})$ converges at a stage s then at the stage when $\varphi_i(s)$, where $i \leq l+1$, converges there will be a guard enumerated into A_p.

Now consider the definition of the function $\beta(q,k)$. The commitment has been made that after the incomparability marker n^q_k has been enumerated into A_q, any element less than n^q_k that enters an A_p such that $p \mid_P q$ results in an n^q_k guard entering A_q. That being the case, the sum

$$\sum_{\substack{l < Mk+q, \\ \text{and } p \mid_P q}} [\alpha(p,l)(h(p,l)+1)]$$

counts the number of guards required due to the domination markers of higher priority in A_q and their associated guards. Additionally, as the incomparability markers $n^p_{k'}$ have priority over n^q_k for $Mk' + p < Mk + q$ the sum

$$\sum_{\substack{Mk'+p<Mk+q,\\ \text{and } p \mid_P q}} \beta(p, k')$$

counts the guards needed to account for the entrants due to these markers. We now show that these functions are finite and in doing so, show that any given marker only requires a finite number of guards.

Lemma 4.12. *The functions $\alpha(p, l)$ and $\beta(q, k)$ are finite.*

Proof. If p is a minimal element then A_p will not need to strong settling time dominate another set. That being the case, Γ^p_0 does not require guards which gives that $\alpha(p, 0) = 0$. Let q be such that $q \not>_P 1$. With $\alpha(p, 0)$ determined for all minimal p we can inductively compute $\alpha(q, 0)$ since Γ^q_0 only requires guards for the Γ^r_0 such that $q >_P r$. Note that since $q \not>_P 1$ and $q >_P r$, it is the case that $r \not>_P 1$. Now consider $\beta(1, 0)$. Since we have determined $\alpha(q, 0)$ for all q such that $q \not>_P 1$ we know the number of guards that n^1_0 requires. Indeed, n^1_0 only requires guards for those Γ^q_0 such that $q \mid_P 1$ and we know $\alpha(q, 0)$ for all $q \not>_P 1$ and hence for all q such that $q \mid_P 1$. Thus, we may compute $\beta(1, 0)$. With this determined, we can go on to compute $\alpha(p, 0)$ for the remaining $p \in P$.

Continuing inductively, consider $\alpha(p, l)$ and $\beta(q, k)$ where $Mk + q = l + 1$ and suppose that we have determined $\alpha(p', l')$ and $\beta(q', k')$ for all $p' \in P$, $l' < l$, and $Mk' + q' \leq l$. Now if p is a minimal element then $\alpha(p, l) = 0$. Suppose p is such that $p \not>_P q$. Since $p \not>_P q$, A_p does not need to dominate A_q and as such, Γ^p_l does not require guards to account for the entrants of A_q. However, Γ^p_l will require guards for the entrants due to the markers $\Gamma^{q'}_{k'}$ such that $p >_P q'$ and $k' < k$, the number of which we have determined. Additionally, for such a p, Γ^p_l will also require guards for the entrants due to the markers $n^{q'}_{k'}$ such that $p >_P q'$ and $Mk' + q' \leq l$, again, the number of which we have determined by assumption. Lastly, such a Γ^p_l will also require guards for the entrants due to the $\Gamma^{q'}_l$ where $p >_P q'$. Now if $p >_P q'$ and $p \not>_P q$ then we must have that $q' \not>_P q$ as well. Consider the minimal q' such that $p >_P q'$. As mentioned, since q' is minimal we get that $\alpha(q', l) = 0$ and from this we can inductively determine $\alpha(r, l)$ for all $r \in P$ such that $r >_P q'$ and $r \not>_P q$.

Having computed $\alpha(r,l)$ for all $r \not>_P q$, we may determine $\beta(q,k)$. Indeed, n_k^q requires guards for the entrants due to the markers of the form $\Gamma_{l'}^{p'}$ such that $p' \mid_P q$ (and hence $p' \not>_P q$) and $l' < Mk+q = l+1$, the number of which has been determined. Additionally, n_k^q requires guards for the entrants due to the markers of the form $n_{k'}^{p'}$ where $p' \mid_P q$ and $Mk'+p' < Mk+q = l+1$, the number of which has also been determined by assumption. With $\beta(q,k)$ determined we can compute $\alpha(p,l)$ for the remaining $p \in P$ since we have now counted all of the entrants the guards of Γ_l^p will have to account for. Essentially, we have shown that every marker only needs guards to account for the possible entrants due to a finite number of markers which, themselves, only have a finite number of guards and as such, every marker only requires a finite number of guards.

□

With the number of guards for each marker determined we proceed to the actual construction of the c.e. sets $A_1,\ A_2,\ \ldots,$ and A_M.

Construction:

Stage 0: Spread out the markers such that there is enough room for the guards as counted by the functions $\alpha(p,l)$ and $\beta(q,k)$.

Stage $s+1$: If $\psi_j(\Gamma_{l+1}^p[s]) \downarrow$ for some $j \leq l+1$ declare that (p,l,j) has received attention. Additionally, for all $q <_P p$ move the $\Gamma_{l'}^q$ such that $l+1 \leq l'$ and all n_k^q such that $l < Mk+q$ to be greater than $s+1$.

Let a denote the number of previous stages such that (p,l,j) has received attention. Note that $a \leq h(p,l+1)$ since $h(p,l+1)$ bounds the number of times Γ_{l+1}^p moves and hence the number of times $\psi_j(\Gamma_{l+1}^p)$ can converge. Declare that $[i,j,a]_l^p[t]$ enters A_p at the stage t when $\varphi_i(s+1) \downarrow$ where $i \leq l+1$.

Now from previous counting there are at most

$$\sum_{\substack{Mk+q \leq l+1, \\ \text{and } q <_P p}} [(\beta(q,k)+1)(g(q,k)+1)] + \sum_{\substack{k < l+1, \\ \text{and } q <_P p}} [(\alpha(q,k)+1)(h(q,k)+1)]$$

many possible entrants of higher priority in the sets of the form A_q such that $p >_P q$ that are not moved upon the convergence of $\psi_j(\Gamma_{l+1}^p)$.

Let $s_1,\ s_2, \ldots,\ s_U$ denote the stages greater than $s+1$ such that an incomparability marker or guard less than or equal to $s+1$ enters an A_q such that $p >_P q$. Note that

$$U \leq \sum_{\substack{Mk+q \leq l+1, \\ and\ q <_P p}} [(\beta(q,k)+1)(g(q,k)+1)].$$

Declare that at the stage t when $\varphi_i(s_u) \downarrow$ where $i \leq l+1$ and $1 \leq u \leq U$, the Γ_l^p guard

$$[i, j, h(p, l+1)+u]_l^p[t]$$

is enumerated into A_p.

Let $t_1,\ t_2, \ldots,\ t_V$ denote the stages greater than $s+1$ such that a Γ_k^q guard less than or equal to $s+1$ enters an A_q such that $p >_P q$. Note that

$$V \leq \sum_{\substack{k < l+1, \\ and\ q <_P p}} [(\alpha(q,k)+1)(h(q,k)+1)].$$

Declare that at the stage t when $\varphi_i(t_v) \downarrow$ where $i \leq l+1$ and $1 \leq v \leq V$, that the guard

$$[i, j, h(p, l+1) + \sum_{\substack{Mk+q \leq l+1, \\ and\ q <_P p}} [(\beta(q,k)+1)(g(q,k)+1)] + v]_l^p[t]$$

is enumerated into A_p. Notice that $\alpha(p,l)$, indeed, gives the number of guards required by this process.

Now consider the element p such that $p \equiv (s+1)(mod\ M)$. Enumerate into A_p the marker n_k^p such that $s+1 = Mk+p$. Make the commitment that from here on in whenever a number less than or equal to n_k^p enters an A_q such that $p \mid_P q$, an n_k^p guard will enter A_p. Now from previous counting there are at most

$$\sum_{\substack{l < Mk+p, \\ and\ p \mid_P q}} [\alpha(q,l)(h(q,l)+1)] + \sum_{\substack{Mk'+p < Mk+q, \\ and\ p \mid_P q}} \beta(p,k')$$

many possible entrants of the sets, A_q, where $p \mid_P q$, that are less than or equal to n_k^p. Declare that at the stage when the n^{th} such possible entrant enters one of these A_q, the n^{th} guard of n_k^p enters A_p.

This concludes the construction process.

Lemma 4.13. *The $R_{p \mid_P q}$ are satisfied.*

Proof. If $p \mid_P q$ then at every stage s where $q \equiv s \ (mod\ M)$, an n^q_k is placed in A_q such that $s = Mk + q$ with the commitment that whenever a number less than n^q_k enters A_p an n^q_k guard enters A_q. Similarly at every stage t such that $p \equiv t \ (mod\ M)$, an n^p_j is placed in A_p such that $t = Mj + p$ with the commitment that whenever a number less than n^p_j enters A_q an n^p_j guard enters A_p. Now, as we have defined a computable function g bounding the number of times n^q_k and n^p_j may move, these incomparability markers eventually settle. That said, it is the case that $m_{A_q}(n^q_k) \geq m_{A_p}(n^q_k)$ for all n^q_k that have settled where $k \in \omega$. Additionally, we have that $m_{A_p}(n^p_j) \geq m_{A_q}(n^p_j)$ for all n^p_j that have settled where $j \in \omega$. Thus, we have that $(\exists^\infty x)[m_{A_q}(x) \geq m_{A_p}(x)]$ and $(\exists^\infty x)[m_{A_p}(x) \geq m_{A_q}(x)]$ as well, giving that $A_p \not>_{sst} A_q$ and $A_q \not>_{sst} A_p$. Hence, $A_p \mid_{sst} A_q$ as desired.

□

Lemma 4.14. *The $R_{p>_P q}$ are satisfied.*

Proof. Suppose $p >_P q$ and φ_i and ψ_j are total. It suffices to show that for all but finitely many x that whenever a $y \leq \psi_j(x)$ is enumerated into A_q at some stage s then some number less than or equal to x is enumerated into A_p at some later stage $t \geq \varphi_i(s)$. Let x be such that $\Gamma^p_l \leq x \leq \Gamma^p_{l+1}$ for some $l + 1 \geq i, j$ where Γ^p_l and Γ^p_{l+1} have settled. Note that as we had previously defined a computable function h bounding the number of times the domination markers move, we, indeed, have such settled markers, Γ^p_l and Γ^p_{l+1}. Now from our construction process at stage s when $\psi_j(\Gamma^p_{l+1}) \downarrow$ there are Γ^p_l guards assigned with the commitment of entering A_p at the stage when $\varphi_i(t) \downarrow$ where t denotes a stage greater than s such that some entrant less than $\psi_j(\Gamma^p_{l+1})$ enters A_q. That said, whenever a $y \leq \psi_j(x)$ is enumerated into A_q at some stage t, a Γ^p_l guard will enter A_p at the stage when $\varphi_i(t) \downarrow$ since $y \leq \psi_j(x) \leq \psi(\Gamma^p_{l+1})$. This is as what we desire since any Γ^p_l guard is less than Γ^p_l and hence less than x. Additionally, there are guards assigned to enter A_p even if there are no numbers less than $\psi_j(\Gamma^p_{l+1})$ later enumerated into A_q. Thus, we have $A_p >_{sst} A_q$.

□

The previous lemmas give that $A_1,\ A_2, \ldots,$ and A_M are as desired.

□

Notice that the proof above greatly depends on the partial ordering being finite. Indeed, as the partial order is finite there can only be a finite number of incomparable elements and

as such we are essentially able to take turns in performing the actions necessary to preserve the incomparability of each element. That is, at any given stage in the construction process we are only acting to preserve the incomparability $p \mid_P q$ for a fixed element, p, by making the appropriate commitments regarding some marker n^p_k. Since the partial order is finite we are able to do things in such a particular order and can still be assured that every incomparability in the partial ordering is eventually be accounted for.

Similarly, for any given $p \in P$, p can only be in finitely many $<_P$ chains, all of which would be of finite length. The procedure above leans quite heavily on this fact to guarantee that a marker is not moved infinitely many times and only requires finitely many guards. Indeed, the marker Γ^p_l requires guards to account for the possible entrants due to the markers $\Gamma^q_{l'}$ such that $l' \leq l$ and $p >_P q$ and there will only be a finite number of these markers. Additionally, the marker Γ^p_l requires guards to account for the possible entrants due to markers of the form n^q_k such that $p >_P q$ and $Mk + q \leq l + 1$ and since there are only finitely many such markers, Γ^p_l only requires finitely many guards for the sake of these incomparability markers.

Along a similar line, as there are only finitely many elements in the ordering P there are only finitely many markers that can move a given marker Γ^p_l and as seen in the procedure, each movement is finite and only occurs finitely many times. The use of this fact is most apparent in the definition of our functions h, g, α, and β bounding the number of movements and the number of guards of the domination and incomparability markers. Since our ordering is finite, we can guarantee the existence of maximal and minimal elements. Indeed, we know that there are finitely many. This fact, essentially, provides us with a starting point to build our functions. As can be seen in the inductive definition of the function h, a maximal element is needed in the base case from which we could go on to define h on the other markers. Similarly, in the inductive definition of the function α, we appeal to the existence of minimal elements to establish a base case to grow from. Now in the setting of an infinite partial ordering, we can no longer guarantee the existence of minimal or maximal elements and as such, there is no obvious starting point from where we can start building our functions and hence our c.e. sets.

The explanation above provides one of several difficulties that comes along with trying to embed a countably infinite partial ordering into $\mathcal{E}_{sst}$. Some of these issues are the result of not being able to maintain such a rigid control of where the markers are placed with regards to each other and how they interact. Indeed, we are no longer able to take turns

with the incomparability markers as is done in the finite case but at the same time we still must ensure that the appropriate actions are eventually performed for these markers. In addition to this, one must be cautious in deciding the interaction of the markers since the domination and incomparability markers *play off* one another which may result in a marker requiring infinitely many guards or never settling. For example, consider the situation where there are markers arranged such that $\Gamma_l^p < n_k^q < n_k^r < \Gamma_{l+1}^p$ where $p >_P r$, $r \mid_P q$, and $p \mid_P q$. Now if we were to assign guards to Γ_l^p to account for the entrants due to n_k^r as is done in the finite case then there would end up being a *loop* where Γ_l^p would require infinitely many guards. Indeed, there would be the situation where Γ_l^p would require guards for n_k^r which would require guards for n_k^q and in turn, n_k^q would require guards for Γ_l^p. Essentially, Γ_l^p would require guards to account for its own guards. Furthermore, since the domination markers can cause the incomparability markers to move and vice versa, one must take caution to ensure that the markers all eventually settle. In particular, we do not want the situation where the movements caused by a marker force the marker, itself, to move.

To embed a countably infinite partial ordering one cannot be as particular as in the proof of Theorem 4.9, the success of which relies quite heavily on the partial order being finite, but at the same time avoid some of the pitfalls noted above. Nevertheless, there has been no indication that such embeddings cannot be possible. Indeed, the fact that any finite partial ordering and the linear orderings from Theorem 4.8 can be embedded into $\mathcal{E}_{sst}$ support the possibility of generalizing Theorem 4.9 to the countably infinite case. That said, we close this thesis with the following conjecture.

Conjecture 4.15. Any countable partial order can be embedded into $\mathcal{E}_{sst}$.

Bibliography

[1] Barry S. Cooper. *Computability Theory.* CRC Press, Inc., Boca Raton, FL, USA, 2002.

[2] Barbara F. Csima. The settling time reducibility ordering and Δ^0_2 sets. *J. Logic Comput.*, 19(1):145–150, 2009.

[3] Barbara F. Csima and Richard A. Shore. The settling-time reducibility ordering. *J. Symbolic Logic*, 72(3):1055–1071, 2007.

[4] Barbara F. Csima and Robert I. Soare. Computability results used in differential geometry. *J. Symbolic Logic*, 71(4):1394–1410, 2006.

[5] Alexander Nabutovsky and Shmuel Weinberger. The fractal nature of Riem/Diff. I. *Geom. Dedicata*, 101:1–54, 2003.

[6] Robert I. Soare. *Recursively enumerable sets and degrees.* Springer-Verlag New York, Inc., New York, NY, USA, 1987.