Dynamic Isolation for Protection in the Movable Feast Machine

James Vickers
jvick3@unm.edu
University of New Mexico
Albuquerque, NM, USA

Abstract

The *Movable Feast Machine* (MFM) is a computational platform which encourages robustness and scalability by limiting the physical area of reads and writes by its *Elements* at any one time. Even with this spatial access restriction, writes performed by these Elements can have adverse effects to important computations. We present a *Dynamic Isolator* to robustly separate diffusing *Atoms*, proving very useful for prolonging their existence and integrity. The Dynamic Isolator provides strong protection against destructive Elements via a simple and general-purpose spatial separation algorithm.

Introduction

The platform for this work is the Movable Feast Machine (MFM). The MFM is a robust spatial computer architecture that could be described as a kind of asynchronous *Cellular Automata* (Ackley et al., 2012). In the MFM programming model, computation is done via the behavior of *Elements*. At any given discrete *Site* in the MFM, there exists an instance of a particular Element, called an *Atom*. At initialization time, every Site has the *Empty* Atom, which has no behavior and holds no data.

The basic operation of the MFM is to randomly choose a Site, determine the Element type of the Atom currently at that Site, and call it to act. An Atom acts when the *Behavior Function* for its Element type is called, which has a reference to a *Event Window*. An Event Window is a Moore Neighborhood of Sites at Manhattan Distance of *Event Radius* or less from the Atom. When an Atom is called to act, this is known as an *Event* for the Atom that is of that type. In this paper, all experiments are performed using an MFM simulator, developed by Professor David Ackley and Trent Small of the University of New Mexico. Figure 1 shows the architecture of the MFM. For more detailed information on the MFM, see *A Movable Architecture for Robust Spatial Computing* (Ackley et al., 2012).

In the MFM, an Element can read anything and write whatever will fit on any Site in its Event Window when called to act. It could copy itself to every location in the Event Window (a powerful version of what is known as a *Fork Bomb*, which rapidly occupies all available Site's with

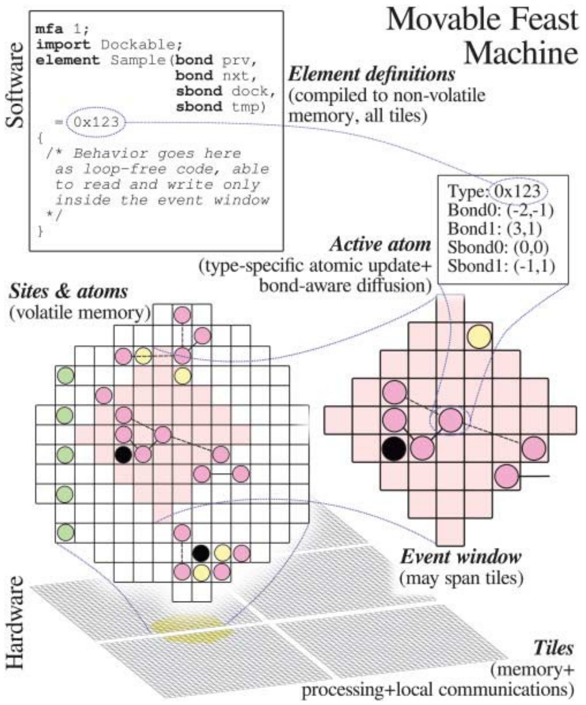

```
mfa 1;
import Dockable;
element Sample(bond prv,
               bond nxt,
               sbond dock,
               sbond tmp)
  =(0x123)
{
 /* Behavior goes here
   as loop-free code, able
   to read and write only
   inside the event window
  */
}
```

Figure 1: Architecture of the MFM, taken from *A Movable Architecture for Robust Spatial Computing* (Ackley et al., 2012).

itself) or erase anything it doesn't like. Common activities for MFM Elements during an Event include spatial diffusion and reproduction of Atoms. An Atom may also swap an Atom to another Site in its Event Window. Note that there is no concept of read-write privilege in the MFM; any Site within the Event Window of an Atom is accessible to it. What then, stops precious data in the system from being destroyed or corrupted on a wide scale by malicious or malfunctioning Elements?

The purpose of this work is to help mitigate this issue. I present a new Element, *Dynamic Isolator*, which serves to protect Atoms by spatially separating them from each other. The design goal of Dynamic Isolator is to keep any two Atoms not of type Empty or Dynamic Isolator from being within Event Radius of each other. It does so to prevent these Atoms from having access to each other in the Event Window when their Behavior Function is called. Dynamic Isolator does so by i) surrounding other Atoms with an *Isolator Bubble* of Dynamic Isolator Atoms and ii) Swapping the Atom inside an Isolator Bubble away from itself when other Atoms are seen, leading to *Bubble Repulsion*. Combined, these two activites allow Dynamic Isolator to wrap Atoms in a layer of Dynamic Isolator, follow Atoms as they diffuse, and maintain that Isolator Bubbles only contain one Atom within them. Dynamic Isolator's do not reproduce, and destroy themselves if they do not see an Atom to protect during an Event.

To show the capabilities of Dynamic Isolator, I pit it against a destructive Element called *Eraser*. Eraser does like its name suggests, and erases Sites that are within a configurable distance of it. The Element that Dynamic Isolator tries to safeguard against Data in this paper is *Data*, a storage Element that does not reproduce and diffuses randomly during each Event it recieves. Data was used as part of *Demon Horde Sort* in previous work on the MFM (Ackley et al., 2011).

The Dynamic Isolator Model

The Dynamic Isolator seeks to surround other Atoms with itself, and move those Atoms away from each other. The design goal of Dynamic Isolator is to have Atoms that are not of type Dynamic Isolator or Empty be within the same Event Window. The *Inner Radius* parameter of Dynamic Isolator determines how much Empty space is maintained between an Atom and the Dynamic Isolator's surrounding it. If a Dynamic Isolator does not see any Atoms with a type other than its own or Empty, it will erase itself. Dynamic Isolator has two primary behaviors:

Behavior 1: Surround other Elements: When a Dynamic Isolator at Site A sees any non-Empty Atom that is not of its own type at Site B, it will look at every Site C that is within Event Window Radius R of A in Manhattan Distance:

- If C is less than Inner Radius away from B and holds a Dynamic Isolator, set C as Empty
- If C is at least Inner Radius away from B and is empty, set C as Dynamic Isolator

This behavior is called *Bubble Construction* and is how Isolator Bubbles are formed around Atoms and maintained as the Atom diffuses spatially. The distance between the center Atom and the Dynamic Isolator's is generally constant at Inner Radius. This add-and-erase behavior is how Dynamic Isolator's are collectively able to keep an Atom surrounded without leaving remnants as it diffuses spatially.

Behavior 2: Swap Elements apart: Assume a Dynamic Isolator at Site A sees a non-Empty Atom that is not of its own type at Site B, and that the Manhattan Distance between Sites A and B is denoted by *r*. In this situation, Dynamic Isolator will look at every Site C that is of Manhattan Distance greater than r from B. If the Atom at C is not of type Dynamic Isolator nor Empty:

Let X be the horizontal offset from A to B, and Y be the vertical offset from A to B.

- If X is 0, swap B one unit in Y direction away from A
- If Y is 0, swap B one unit in X direction away from A
- If both X and Y are non-zero, randomly choose to swap B one unit in either the X or Y direction away from A

With this behavior, Dynamic Isolator's will repel (swap away) an Atom P if it also sees some other Atom P' further away from itself than P. This can cause *Bubble Repulsion* between groups of Dynamic Isolator's surrounding different Atoms, as well as break up Atoms that are already close together when Dynamic Isolator's surround them. Figure 2 shows an overview of the Dynamic Isolator model.

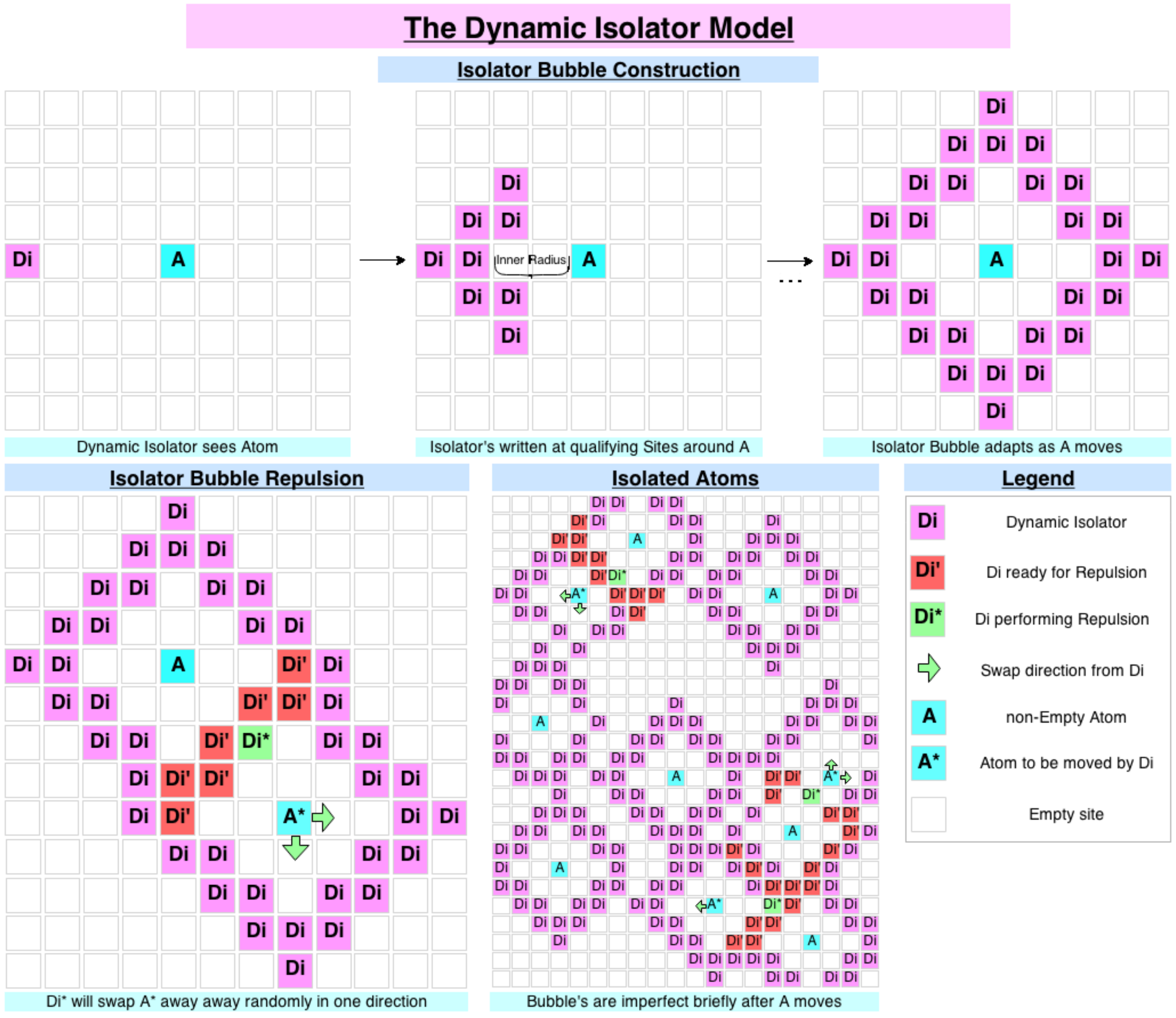
The Adversary: Eraser

The *Eraser* Element is a simple brute designed to test the performance of Dynamic Isolator. One could imagine Eraser as a malicious Element, seeking to destroy information and computational systems in the MFM, or as some kind of rogue Element that is malfunctioning due to data corruption or a mistake in design or implementation. It takes one Element parameter, *Erase Distance*, with possible values $\{1,2,3,4\}$. Its behavior is very simple. When called to act, Eraser will:

- Set any Site at Manhattan Distance Erase Distance or less from itself as *Empty*
- Diffuse randomly by one Site

Eraser does not care about the type of Atom present at the Sites it chooses to erase, which means it will also remove other Eraser's. Eraser does not reproduce. A population of Eraser's in a given MFM environment with no other destructive Atoms will eventually destroy each other and whatever other Atoms are present, but leave a single Eraser remaining (a *Highlander Eraser*). When Erase Distance is set to 1, Eraser will wipe any Site that is directly adjacent to it (Manhattan distance 1). At its maximum value in this simulator, 4, Eraser will remove everything in its Event Window except for itself, as the Event Radius is set to 4.

Eraser could be viewed as the MFM equivalent of a faulty or malicious program on a normal (Von Neumann) computer that removes anything it has access to. In a normal computer, this would generally be anything the program has



The Dynamic Isolator Model
Isolator Bubble Construction
Dynamic Isolator sees Atom
Isolator's written at qualifying Sites around A
Isolator Bubble adapts as A moves
Inner Radius
Isolator Bubble Repulsion
Di* will swap A* away away randomly in one direction
Isolated Atoms
Bubble's are imperfect briefly after A moves
Legend
Di Dynamic Isolator
Di' Di ready for Repulsion
Di* Di performing Repulsion
Swap direction from Di
A non-Empty Atom
A* Atom to be moved by Di
Empty site

Figure 2: The Dynamic Isolator Model.

write-permission on from the Operating System, with its effectiveness limited by the runtime and access permissions of the program. In the MFM, Eraser can remove Atoms that are within Erase Distance of it during an Event, with its overall effectiveness limited by the Eraser population, Eraser Distance, and the spatial layout of other Atoms in time. There is no concept nor mechanisms for access permissions on Sites nor Atoms in the MFM; it would go against its asynchronous, de-centralized foundations.

Implementation

Initial attempts at designing and implementing Dynamic Isolator failed in interesting ways. A naive first design of Bubble Construction had Dynamic Isolator write to all Sites at distance Inner Radius around any Element, but never delete Dynamic Isolator's, and die with a probability configurable by Element parameter. Under this setup, Elements are indeed surrounded by Dynamic Isolator's and followed as they diffuse, but Bubble Repulsion was not a concept at this point and so no real protection was offered. Not erasing Dynamic Isolator's closer than the desired Inner Radius to an Element, combined with dying with a probability rather than dying when no Element is currently seen, caused excessive Dynamic Isolator's to wander around the MFM, with no visible Element to protect.

After fixing and implementing the Bubble Construction algorithm and changing Dynamic Isolator to immediately die when no other Elements not of its own type are visible, simple and lean Isolator Bubbles would form. This was encouraging, but it was observed that nothing stopped Isolator Bubbles from colliding into each other. Thus, they were mostly for show, as an Element within such a Bubble could move into the Event Window of any other Element as usual.

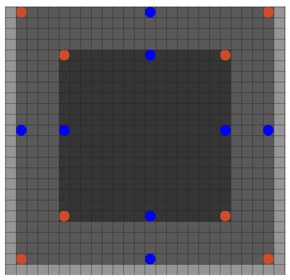
The lack of protection given by non-repulsing Isolator Bubbles made it clear that the merging of those Bubbles should be prevented if Elements are to be afforded real protection. The concept is to have Isolator Bubbles avoid combining with one another; that is, the Bubbles should repel each other in some way. As a first attempt at solving this problem, Dynamic Isolator's would look at Sites further from themselves (and thus in the opposite direction of that Element) than the Atom they saw. If they saw other Dynamic Isolator's at those Sites, they tried to determine if they were part of their own Bubble, or the edge of another one. This was done by looking at the distance of the seen Dynamic Isolator's in comparison to the acting one. If a Dynamic Isolator believed it saw another Isolator Bubble nearby that was not part of its own, it would move its Element away from itself, similarly to how its done currently.

Every attempt at trying to make Dynamic Isolator's repel when they saw each other, rather than when they saw other Atoms, ended in instability and failure to actually achieve repulsion. I believe the problem with this approach lies in the fact that the Atom surrounded by Dynamic Isolator's can move, perhaps multiple times, before the Dynamic Isolator's around it can get an Event of their own to react. Thus, it seems common that a Dynamic Isolator frequently sees others of its own type that could be viewed as another Isolator Bubble. This causes Dynamic Isolator's to move the Atom at the center of the Bubble very frequently from false alarms. This Atom would get moved wildly all over the grid by the Dynamic Isolator's, and ends up in a corner of the MFM. This approach also does not achieve actual repulsion, as Dynamic Isolator's can incidentally move Atoms towards another based solely on a belief that another Isolator Bubble is nearby.

This trial-and-error process led to the development and implementation of the Dynamic Isolator as it exists currently. Its add-and-erase behavior on Atoms of its own types allows it to surround Elements without leaving excess copies of itself around. The Bubble Repulsion algorithm does its best to keep Isolator Bubbles from combining, which helps prevent any two Atoms at the center of such a Bubble from being visible (and thus potentially vulnerable) to each other. The fact that Dynamic Isolator is completely agnostic to the types of Elements it isolates means it can protect both good and bad Elements. Dynamic Isolator does not use the bits allocated to instance Atoms (71 bits in this MFM simulator) for anything, as it maintains no internal state of any kind.

Experiments

There are two primary measurements in the experiments to show the performance of Dynamic Isolator in preventing Eraser's from destroying other Atoms. First, the population of Data, Eraser, and (when applicable) Dynamic Isolator are measured in time. Second, I define a measure called *half-life*, which is the amount of time in *AEPS* at which half of the initial population of an Element is deleted. AEPS, or Average Events Per Site, is the time metric used for the MFM. It denotes the average number of Event's that have occurred at each Site in the MFM up to some instant in time (Ackley et al., 2012). A related term is *kAEPS*, which is 1000 AEPS.


(a) Experiment 1

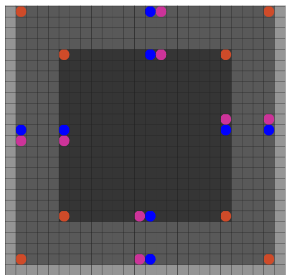

(b) Experiment 2

Figure 3: Orange Sites have an Eraser (arranged in an X-shape), blue Sites have a Data (arranged in a cross-shape), and purple Sites next to Data have a Dynamic Isolator.

Experiment 1: Data and Eraser

As a base case, equal amounts of Data and Eraser Atoms are placed across the grid. The pattern is per-tile in the MFM, and is depicted in Figure 3a The setup is intended to be as symmetrical as possible, and has no Eraser within an Event Window of any other non-Empty Site initially. The Eraser Distance parameter is varied from 1 to 4, with separate runs that are identical in setup but for the value of that parameter.

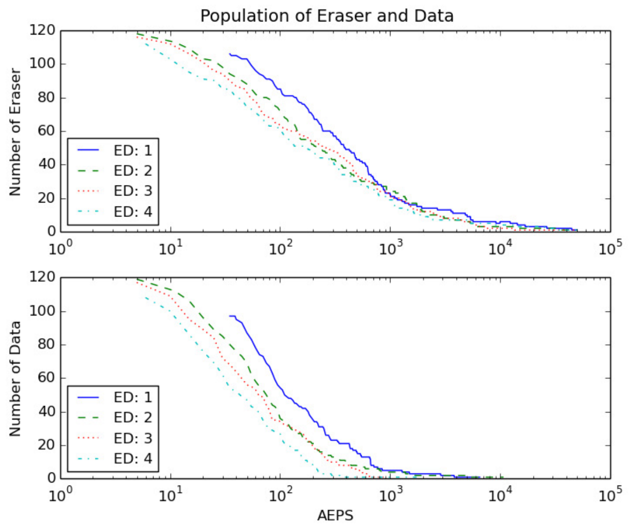


Figure 4: Population of Eraser and Data over time. The horizontal axis (AEPS) is logarithmic. ED indicates the value of the Eraser Distance parameter.

The MFM simulator is run with a set of 5x3 tiles for 50 kAEPS. With 8 each of Data and Eraser per tile and 15 tiles, there are 120 Eraser and 120 Data at the start of the experiment. Counts of Eraser and Data Atoms are recorded. The results are shown in Figure 4.

Experiment 2: Data, Eraser, and Dynamic Isolator

This experiment has a similar setup to Experiment 1, except that a Dynamic Isolator is placed adjacent to each Data Atom as shown in Figure 3b. Once again, their placement is intended to be as symmetrical as possible.

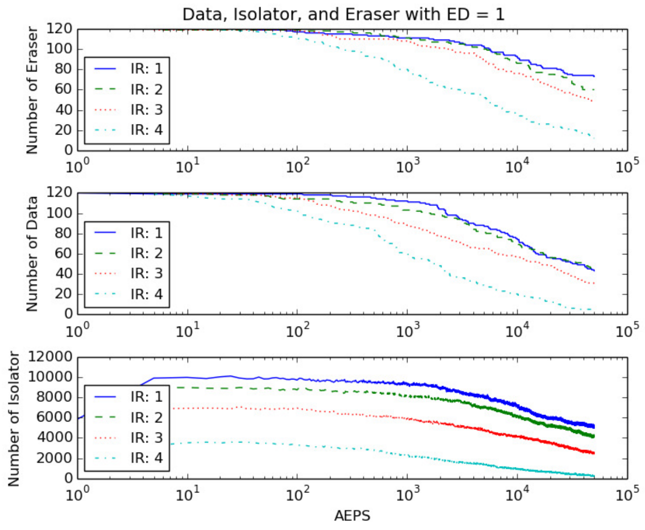


Figure 5: Population of Eraser, Data, and Dynamic Isolator over time with an Eraser Distance of 1. The horizontal axis (AEPS) is logarithmic. IR indicates the value of the Inner Radius parameter for all Dynamic Isolator's.

As in Experiment 1, a 5x3 grid is used and the entire system is run for 50 kAEPS. Separate experimental runs were performed for each combination of Eraser Distance 1,2,3,4 and Inner Radius 1,2,3,4, for a total of 12 trials. The results are shown in Table 1 and Figures 5-8.

Experiment 3: Dreg and Dynamic Isolator

The Dynamic Regulator, or *Dreg*, is an Element of the MFM mentioned frequently in previous works (Ackley et al., 2012, 2011). The main function of Dreg is the management of *Occupied Site Density*, or basically how many non-Empty Atoms are in the MFM. When Dreg gets an Event, it inspects a random Site in its Event Window adjacent to itself. If the inspected Site is Empty, Dreg may place a *Res* ('Resource') Atom there with some probability or it may place another Dreg there with a different probability. If the inspected Site is not Empty, Dreg will erase it with configurable probabilities for the cases if the Site holds another Dreg or a different Element. When one Dreg is placed in the MFM, it fills space

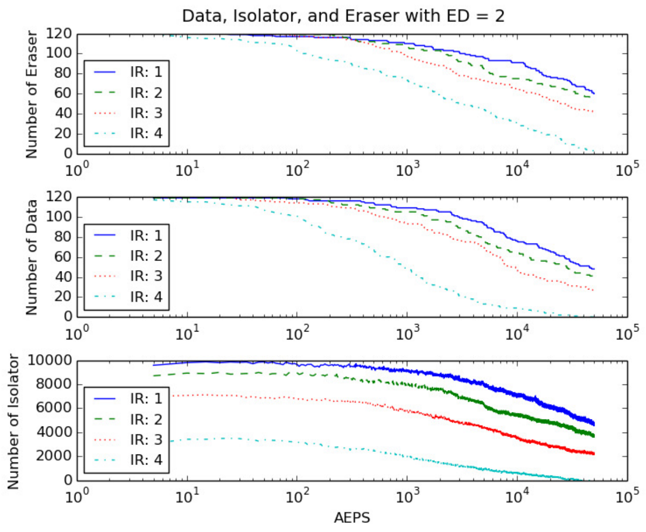


Figure 6: Population of Eraser, Data, and Dynamic Isolator over time with an Eraser Distance of 2. The horizontal axis (AEPS) is logarithmic. IR indicates the value of the Inner Radius parameter for all Dynamic Isolator's.

with Dreg and Res to densities determined by the destruction and creation odds just mentioned. Both Dreg and Res diffuse, but Res does not reproduce. For more info on Dreg, see *A Movable Architecture for Robust Spatial Computing* (Ackley et al., 2012).

As a base case, a single Dreg is placed into the center of the MFM simulator and the system is run for 50 kAEPS. It has these probabilites:

- Create Res in Empty Site: 1 in 200
- Create Dreg in Empty Site: 1 in 500
- Delete non-Dreg Atom: 1 in 100
- Delete Dreg Atom: 1 in 50

The population of Dreg and Res Atoms are measured over 50 kAEPS.

To observe the effects of Dynamic Isolator on this system, one Dynamic Isolator is placed two Sites away from the Dreg. The reason for not placing the Dynamic Isolator adjacent to the lone Dreg is to prevent the possibility that the Dreg deletes the only Dynamic Isolator upon its first Event. As in the base case, the system is run for 50 kAEPS; this time the populations of Dreg, Res, and Dynamic Isolator are measured. Figure 9 shows the results.

Discussion

With no protection from Dynamic Isolator's in Experiment 1, the population of both Data and Eraser is decimated, eventually leaving zero Data and a single Eraser as the stable state. The population of Eraser generally lives longer than

	No Isolator		IR = 1		IR = 2		IR = 3		IR = 4	
ED	Data	Eraser	Data	Eraser	Data	Eraser	Data	Eraser	Data	Eraser
1	90	245	14735	*(F:73)*	17155	39245	6705	23470	905	2575
2	60	145	26785	49235	12130	33735	6910	13810	635	1810
3	46	120	49825	7340	*(F:65)*	5010	2970	2025	495	545
4	35	105	*(F:92)*	220	*(F:85)*	245	16150	180	205	102

Table 1: Half-life with and without Dynamic Isolator. Half-life is measured in AEPS. IR is the Inner Radius of Dynamic Isolator and ED is the Erase Distance. *(F:#)* indicates that the population of that Element did not halve (fall from 120 down to 60) during that experiment and gives the final population of that Element at 50 kAEPS.

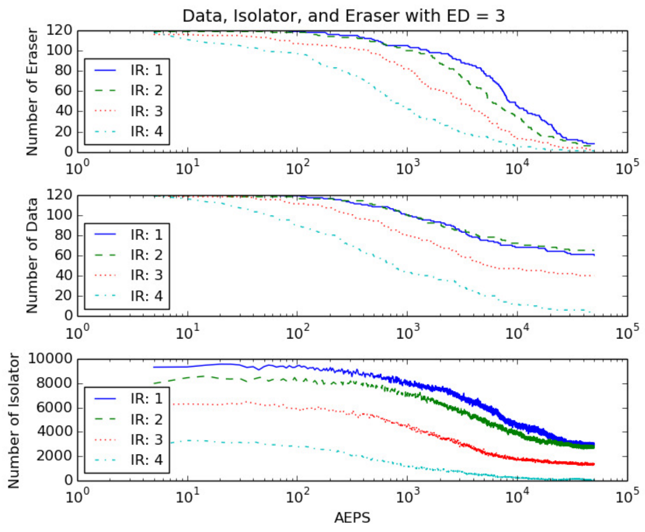


Figure 7: Population of Eraser, Data, and Dynamic Isolator over time with an Eraser Distance of 3. The horizontal axis (AEPS) is logarithmic. IR indicates the value of the Inner Radius parameter for all Dynamic Isolator's.

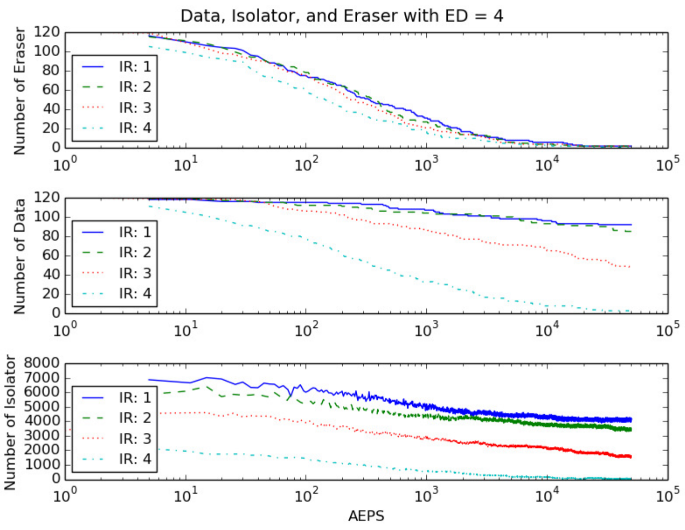


Figure 8: Population of Eraser, Data, and Dynamic Isolator over time with an Eraser Distance of 4. The horizontal axis (AEPS) is logarithmic. IR indicates the value of the Inner Radius parameter for all Dynamic Isolator's.

Data. When an Eraser encounters another Eraser, it can possibly delete it before dying itself. Conversely, when Data meets Eraser it is defenseless in this setup. The half-life of both Eraser and Data are inversely related to the Eraser Distance parameter as can be seen from Table 1. From Figure 4, it appears that varying the Eraser Distance varies the position of the declining curve of population but not necessary the slope of that curve. A high Eraser Distance leads to a quick decline in population of each Element, but after that time the grid is less dense and the effective range of Eraser is not as much as a factor.

Experiment 2 shows how Dynamic Isolator can provide protection to diffusing Elements, both destructive like Eraser and non-destructive like Data. Figure 6 shows the populations of Data, Eraser, and Dynamic Isolator when Eraser Distance is set to 2. The low value of Inner Radius, two, generally provides the best protection but also requires the most Dynamic Isolator's. An Inner Radius of three provides a similar level of protection, but with a significantly lower amount population of Dynamic Isolator's; three represents a kind of 'sweet spot' for the Inner Radius. Raising the Inner Radius up to its maximum value of four gives the thinnest Isolator Bubbles and thus the weakest protection. At this setting, the Isolator Bubble is one unit thick, and even non-destructive Elements like Data can (and occasionally do) escape their Bubble by moving multiple times in the same direction before the Dynamic Isolator's can follow.

Figure 8 shows an interesting case from Experiment 2 in which the Eraser Distance is set to its maximum value of four. At this setting, an Isolator Bubble can only be formed around Eraser until it gets an Event, which immediately destroys that Bubble. Due to this, the population of Eraser's falls much faster than that of Data, when it is normally the opposite. At around 10 kAEPS in the graph, the population of Eraser's has mostly destroyed each other, allowing the population of Data to roughly stabilize.

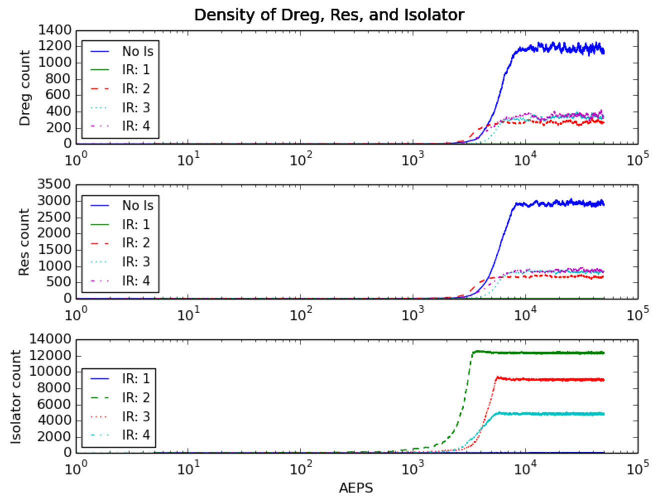


Figure 9: Population of Dreg, Res, and Dynamic Isolator over 50 kAEPS both starting with one Dreg and one Dynamic Isolator and one Dreg only.

The full listing of half-life's can be seen in Table 1. For low-power Eraser's with Eraser Distance of one, Dynamic Isolator with an Inner Radius of two can provide excellent protection to both Data and Eraser, increasing their half-life's by factors of 190.6 and 160.2, respectively. The half-life of Eraser and Data generally decrease as the Inner Radius increased and the Eraser Distance is held constant. For high-powered Eraser's and Dynamic Isolator's with close protection, the half-life of Data can extend beyond the 50 kAEPS of the experiment runs.

In the base case (single Dreg) of Experiment 3, the population of both Dreg and Res rise up and then fluctuate near a constant population of each. The 5x3 grid used in this MFM simulator has 15,360 Sites in it. Figure 9 illustrates that the stable population values appear to correspond to around an 8% density for Dreg and 20% for Res.

Figure 9 shows the population dynamic between Dreg, Res, and Dynamic Isolator. The shape of the curves look very similar to those of Figure 9, but the stable density values for Dreg and Res are considerably lower, depending on the value of Inner Radius. The number of Dynamic Isolator's grows to be massive as the Inner Radius is decreased. At an Inner Radius of 2, Dynamic Isolator comes to occupy around 85% of the grid. Once Dynamic Isolator reaches this level of population, its numbers flucuate little, if at all. In a dense environment, a Dynamic Isolator will usually always see at least one other Element, and thus will not delete itself. With an Inner Radius of 1, Dreg normally only sees Dynamic Isolator's in its Event Window, and very rarely will see an Empty site at which to place a Res or another Dreg. When ran for 50 kAEPS with Inner Radius 1 and the Dreg and Res spawn probabilities described, the population of Dreg remained at one and the population of Res at zero. If the spawn probabilities of Dreg and Res are maximized (made 1-in-1), Dreg can occasionally spawn a Res or another Dreg, which gets pushed into its own Isolator Bubble. Dynamic Isolator is perhaps not well-suited with Dreg, as they are naturally at odds; Dreg seeks to fill the grid to some density of Res and Dreg, while Dynamic Isolator would like every Atom to be separated and thus prefers low-density environments. With an Element like Dreg, Dynamic Isolator functions as less of a protector and more like litter in the environment.

Dynamic Isolator with non-diffusing Atoms

In every example use of Dynamic Isolator so far, it has been used to surround and protect Atoms from each other *as they diffuse*. What about Atoms that don't diffuse? Examples of these stationary Elements currently implemented in the MFM include *Block* and *Wall*. These Elements do not diffuse on their own, but as per the operations of the MFM can be relocated by other Atoms. What happens to Atoms like this around Dynamic Isolator's?

If a Dynamic Isolator is placed nearby (within an Event Window) such a non-diffusing Atom, it will surround it with an Isolator Bubble as usual. Since the Atom at the center of the Bubble does not diffuse, this configuration is stable in the absence perturbations from other Atoms. As long as an Element like Block is surrounded by Dynamic Isolator, does not diffuse, and no other Elements intervene, no Dynamic Isolator's are created nor destroyed after the initial Isolator Bubble is constructed. If in the same environment, diffusing Elements such as Data are added, these end up in an Isolator Bubble as they diffuse close to a stationary Isolator Bubble. When such a moving Isolator Bubble gets close to one with a stationary Element, the Dynamic Isolator's do their Bubble Repulsion to the stationary Element and it gets moved away from the nearby diffusing Element. The long-term result of such a system is that the stationary Isolator Bubbles get pushed into the edges, and eventually the corners, of the MFM environment they are in.

What about when a Dynamic Isolator is placed near a dense cluster of non-diffusing Atoms? Dynamic Isolator's will be written to Sites around the cluster, but then the Bubble Repulsion action of Dynamic Isolator will break apart this cluster into separated Isolator Bubbles, most often with a single Element at the center of each. Some Isolator Bubbles may end up with multiple (usually no more than two) Atoms at their approximate center, due to a back-and-forth behavior of Dynamic Isolators that end up equally and directly opposed to each other in the final spatial layout.

Future work

One possible area of improvement for Dynamic Isolator would be to make it less tolerant of abuse from destructive Elements such as Eraser. As Dynamic Isolator exists to protect the existence of Atoms, regardless of type, it seems perhaps ill-fitting for it to ever destroy one. However, one could imagine a policy where Dynamic Isolator's check on the number of each other visible in an Isolator Bubble. If some kind of attack is detected, say a sharp reduction in the number of Dynamic Isolator's, they could respond by deleting any Element in the Event Window that is not of their own type and is present in at least two Sites.

One potential issue with Dynamic Isolator is that it severely limits the ability of Atoms to interact with each other. While this is done by Dynamic Isolator as a very conservative safety policy, it isn't the only policy that could make sense. One could imagine a policy in which alike Atoms (i.e. of the same type) are allowed, or even encouraged, to be close in space, while different Atoms are to be separated.

An Element similar to Dynamic Isolator, call it *Dynamic Aggregator*, could perhaps do just that. If Dynamic Aggregator sees two Atoms of different types during an Event, it could move them in opposite directions of each other. On the other hand, when it sees two Atoms of the same type, it could move them closer together. Dynamic Aggregator would still write itself into bubbles around Elements, just as Dynamic Isolator does. One possible use for an Element like Dynamic Aggregator could be to sort an area in the MFM by Element type.

Conclusion

The Movable Feast Machine presents an interesting and powerful new paradigm for robust-first spatial computing. Its asynchronous nature is not entirely without peril, however; Atoms can destroy or otherwise corrupt anything they see in their Event Window. Dynamic Isolator is presented as a solution to combat this issue when necessary. It does so by surrounding Atoms in a kind of protective bubble, made of itself, and trying to keep those bubbles separated as best as it can. It can provide strong protection to diffusing Atoms from destructive Elements such as Eraser. Its behavior is robust to Atom deletion, as the Isolator Bubble can be rebuilt rapidly in only a few Event's by nearby surviving Dynamic Isolator's. This is illustrated by the observation that an Eraser with an Eraser Distance as high as three can still be surrounded and followed by Dynamic Isolator for long periods. Dynamic Isolator does not use any state across Events for its operations, and thus Dynamic Isolator Atoms are robust from corruption of their internal state.

Dynamic Isolator is an attempt to remedy a fundamental danger in the MFM. The danger is that an ecosystem of computing Atoms could be inadverdently destroyed by a malfunctioning or malicious Element, as shown by the Eraser Element. Dynamic Isolator does its best to mitigate this, but in some instances it can only prolong the inevitable. The asynchronous nature of the MFM means that eventually even a weakly-powered Eraser can get enough Event's in a row to enter an Isolator Bubble and destroy its center Atom. Such an occurrence cannot be fully prevented given the nature of this platform, and this work does not attempt to provide any guarantee on the safety of Atoms. Dynamic Isolator can, however, greatly prolong the survival of Atoms in the presence of destructive ones. Dynamic Isolator represents a kind of spatial separation mechanism, implemented in the Cellular Automata platform of the MFM.

Acknowledgements

I would like to thank Professor David Ackley at the Computer Science Department at the University of New Mexico. I spent many hours in his office discussing ideas and getting feedback on this work during its development. I would also like to thank the fellow members of the Artificial Life research class at that University during the Fall of 2014.

References

Ackley, David H., Cannon, Daniel C., and Williams, Lance R. (2012). *A Movable Architecture for Robust Spatial Computing*. Handling editor Jacob Beal, *The Computer Journal*.

Ackley, D. and Cannon, D. 2011. *Pursue robust indefinite scalability*. The 13th Workshop on Hot Topics in Operating Systems.