MASON Retirement Age

Ernesto Carrella

RA

May 2 2011

Outline

The two souls of the paper

- An experiment in replication

The two souls of the paper

- An experiment in replication
- An experiment in MASON

Replication

- If we choose a very simple model, can we achieve “numerical identity”?

Replication

- If we choose a very simple model, can we achieve “numerical identity”?
- No

Replication

- If we choose a very simple model, can we achieve “numerical identity”?
- No
- Numerical identity is really hard

MASON

- *Models are completely independent from visualization, which can be added, removed, or changed at any time*

MASON

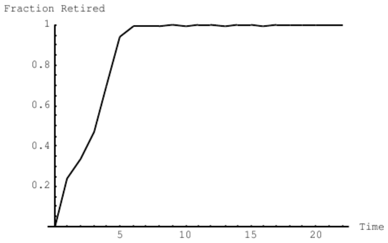
- *Models are completely independent from visualization, which can be added, removed, or changed at any time*
- Can we believe advertisement?

MASON

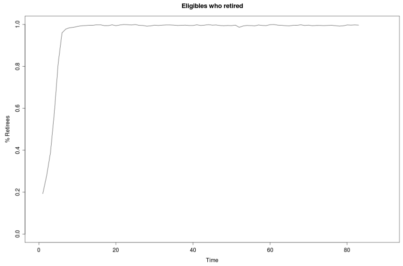
- *Models are completely independent from visualization, which can be added, removed, or changed at any time*
- Can we believe advertisement?
- Maybe.

Outline

The Good



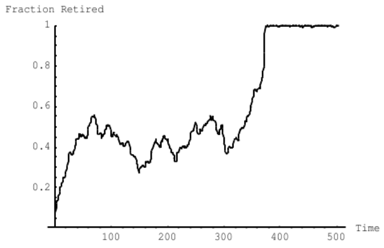
(a) Original Paper



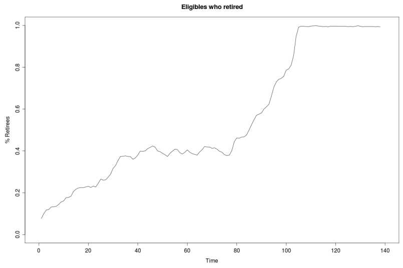
(b) MASON replication

Figure: 20% rational agents case comparison

The Good



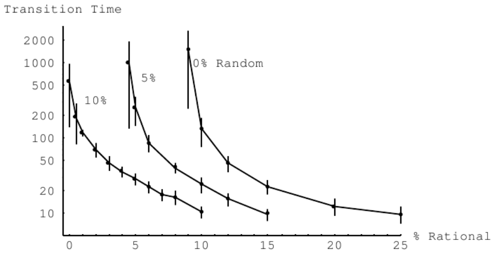
(a) Original Paper



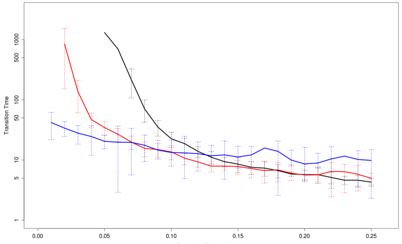
(b) MASON replication

Figure: 5% rational agents case comparison

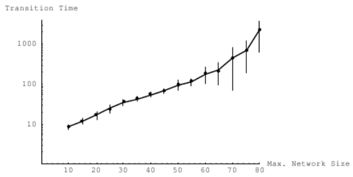
The (kind of) good



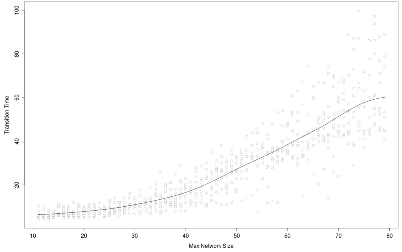
(a) Original Paper



The (kind of) good



(a) Original Paper



(b) MASON replication

Figure: Changes in time to get full-retirement equilibrium by changing maximum network size. Grey dots represent runs

The really bad

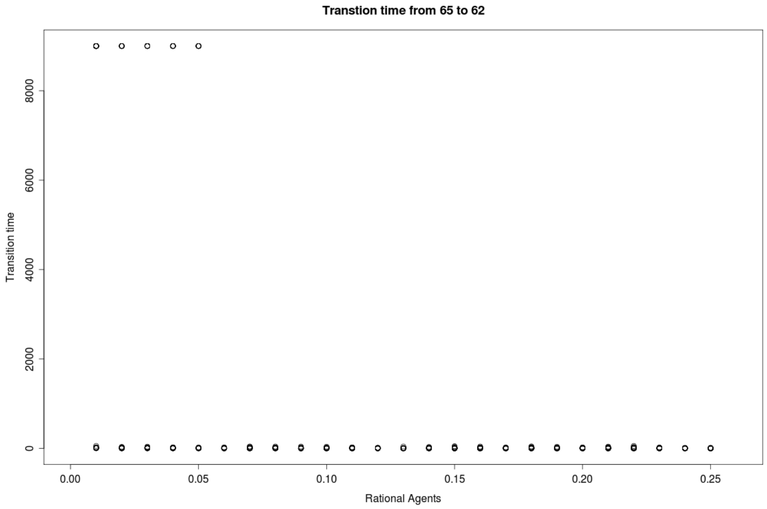


Figure: Weird dynamics

The weird

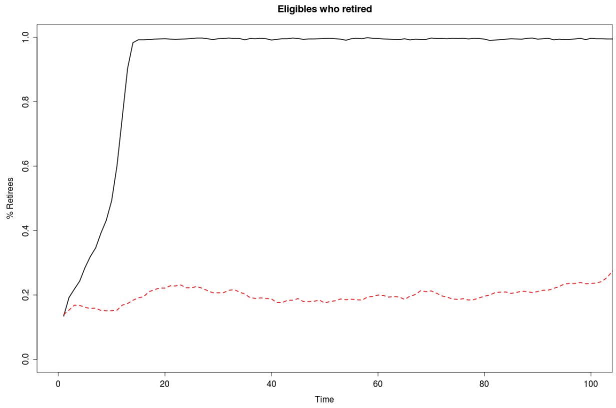


Figure: The black line is the simulation with strict threshold, the red line is without

Outline

The Advantages of Java

- Can use premade data structures
- Can use premade modelling structures
- Object-oriented programming!

Amateur Perspective

- Never write twice

Amateur Perspective

- Never write twice
- Small meaningful functions

Amateur Perspective

- Never write twice
- Small meaningful functions
- Implementation Hiding

Amateur Perspective

- Never write twice
- Small meaningful functions
- Implementation Hiding ... sort of

Agent Type

- Three types of agents

Agent Type

- Three types of agents
  - Rational

Agent Type

- Three types of agents
  - Rational
  - Random

Agent Type

- Three types of agents
  - Rational
  - Random
  - Imitator

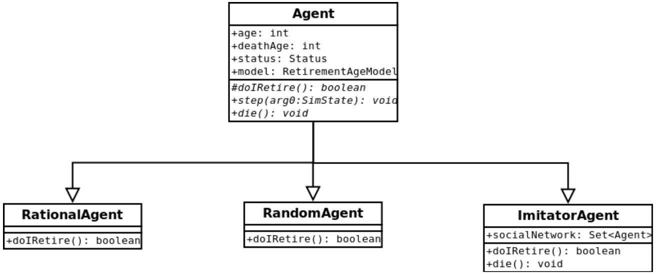
Agent Type

- Three types of agents
  - Rational
  - Random
  - Imitator
- Keep them as a class

Agent Type

- Three types of agents
  - Rational
  - Random
  - Imitator
- Keep them as a class
- Instantiate them in proportion

It's UML Time!


Agent
+age: int
+deathAge: int
+status: Status
+model: RetirementAgeModel
#doIRetire(): boolean
+step(arg0:SimState): void
+die(): void
RationalAgent
+doIRetire(): boolean
RandomAgent
+doIRetire(): boolean
ImitatorAgent
+socialNetwork: Set<Agent>
+doIRetire(): boolean
+die(): void

Agent

- Interface vs Abstract Class

Agent

- Interface vs Abstract Class

```
public void step(SimState arg0) {

age++;

if(age >= deathAge)
this.die();

else if (status == Status.WORKING)
status = doIRetire();
}
```

Overriding

- The difference in agents is only in the method *doIRetire()*

Overriding

- The difference in agents is only in the method *doIRetire()*
- ImitatorAgent extends *die()*

Overriding

- The difference in agents is only in the method *doIRetire()*
- ImitatorAgent extends *die()*
- All we need to do is schedule them.

Outline

The Garbage Collector

- It “automatically” destroys unused objects

The Garbage Collector

- It “automatically” destroys unused objects
- Automatically: unlinked objects

The Garbage Collector

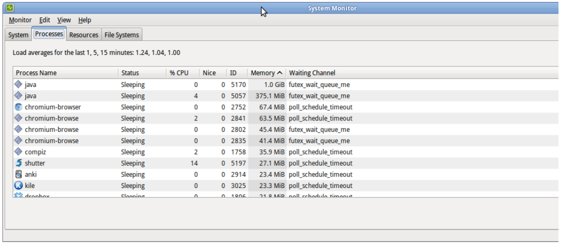
- It “automatically” destroys unused objects
- Automatically: unlinked objects
- Cannot be done manually

Delete agents

- Do we really need to destroy them?

Delete agents

- Do we really need to destroy them?

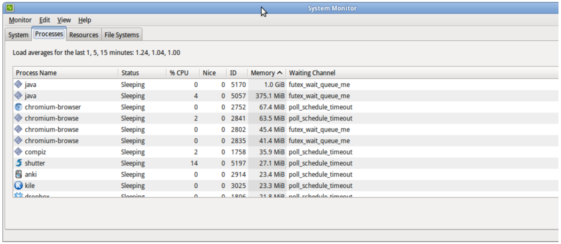


Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

Delete agents

- Do we really need to destroy them?



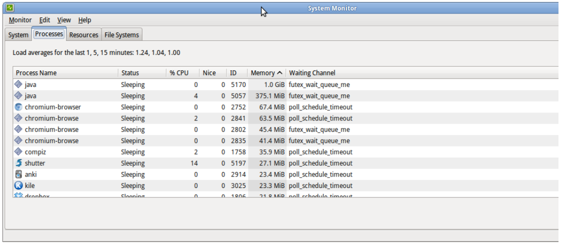
Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes

Delete agents

- Do we really need to destroy them?



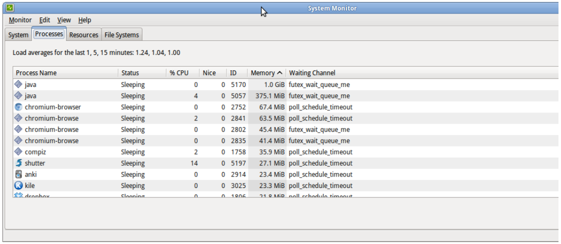
Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes
- What links to them?

Delete agents

- Do we really need to destroy them?



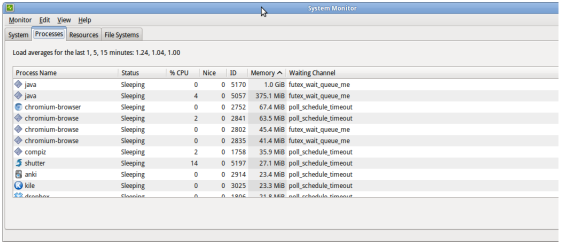
Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes
- What links to them?
  - Simstate's table containing them

Delete agents

- Do we really need to destroy them?



Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes
- What links to them?
  - Simstate's table containing them
  - and that's it?

Schedule and Scheduling

- We schedule each agent separately

Schedule and Scheduling

- We schedule each agent separately
- *Schedule.scheduleRepeating()*

Schedule and Scheduling

- We schedule each agent separately
- *Schedule.scheduleRepeating()*
- How does schedule works?

Schedule and Scheduling

- We schedule each agent separately
- *Schedule.scheduleRepeating()*
- How does schedule works?
- Heap

How to remove a repeating steppable from the heap

- Make the array steppable instead

How to remove a repeating steppable from the heap

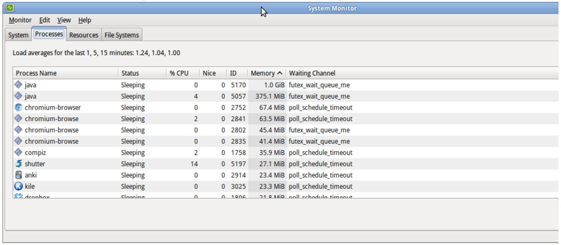
- Make the array steppable instead
- Use Stoppable

How to remove a repeating steppable from the heap

- Make the array steppable instead
- Use Stoppable
- **public** *Stoppable scheduleRepeating(Steppable agent)*

Delete agents 2

- Do we really need to destroy them?



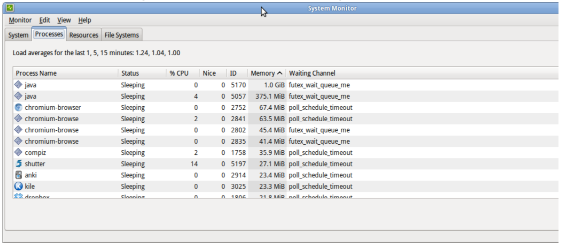
Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes
- What links to them?
  - Simstate's table containing them
  - Schedule

Delete agents 2

- Do we really need to destroy them?



Load averages for the last 1, 5, 15 minutes: 1.24, 1.04, 1.00

Process Name	Status	% CPU	Nice	ID	Memory	Waiting Channel
java	Sleeping	0	0	5170	1.0 GiB	futex_wait_queue_me
java	Sleeping	4	0	5057	375.1 MiB	futex_wait_queue_me
chromium-browser	Sleeping	0	0	2752	67.4 MiB	poll_schedule_timeout
chromium-browse	Sleeping	2	0	2841	63.5 MiB	poll_schedule_timeout
chromium-browse	Sleeping	0	0	2802	45.4 MiB	futex_wait_queue_me
chromium-browse	Sleeping	0	0	2835	41.4 MiB	futex_wait_queue_me
compiz	Sleeping	2	0	1758	35.9 MiB	poll_schedule_timeout
shutter	Sleeping	14	0	5197	27.1 MiB	poll_schedule_timeout
anki	Sleeping	0	0	2914	23.4 MiB	poll_schedule_timeout
kile	Sleeping	0	0	3025	23.3 MiB	poll_schedule_timeout
dropbox	Sleeping	0	0	1806	21.8 MiB	poll_schedule_timeout

- Yes
- What links to them?
  - Simstate's table containing them
  - Schedule
  - and that's it?

Social Network

- A set of Agents

Social Network

- A set of Agents
- These are references to your friends

Social Network

- A set of Agents
- These are references to your friends
- As long as one of your friend is alive (or a friend of that friend...) you will not be recycled

Social Network

- A set of Agents
- These are references to your friends
- As long as one of your friend is alive (or a friend of that friend...) you will not be recycled
- This created enormous slowdowns, even after fixing for the other two

Social Network

- A set of Agents
- These are references to your friends
- As long as one of your friend is alive (or a friend of that friend...) you will not be recycled
- This created enormous slowdowns, even after fixing for the other two
- Morale: Garbage collector thinks you are smarter than you are