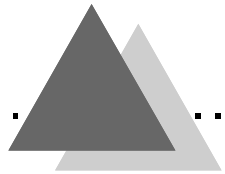
Incremental Mature Garbage Collection Using the Train Algorithm

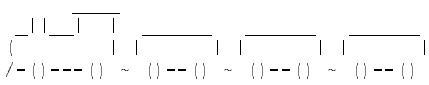
Jacob Seligmann & Steffen Grarup
Aarhus University

August 10th, 1995

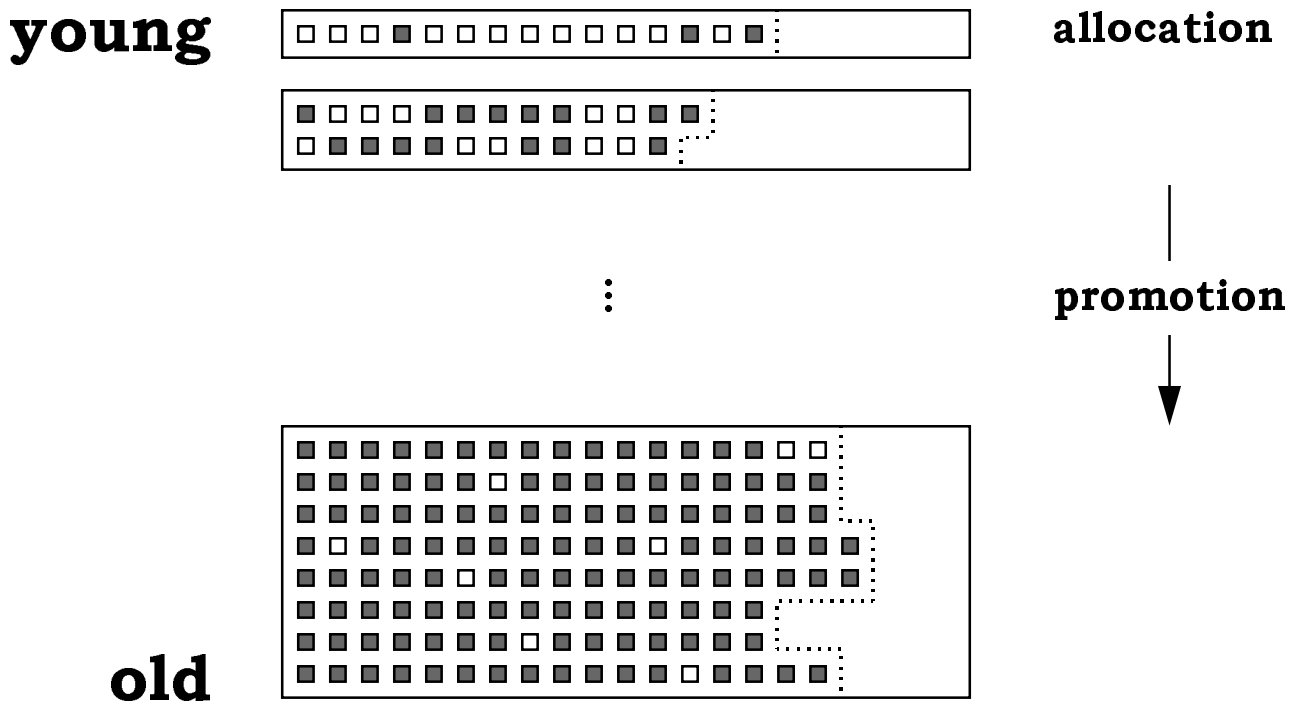
Overview

- Introduction
- The Algorithm
- The Implementation
- The Results
- Discussion





Generation-Based GC

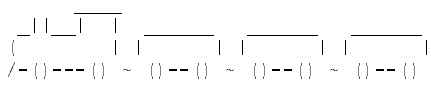


Generation Profile

Generation	Young	Old
Size	small	large
Mortality	high	low
Collection method	copy collection	mark-sweep
Collection frequency	high	low
Collection duration	short	long

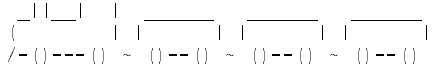
Hudson & Moss

Collection method	mark-sweep	Train Algorithm
Collection frequency	low	medium
Collection duration	long	short

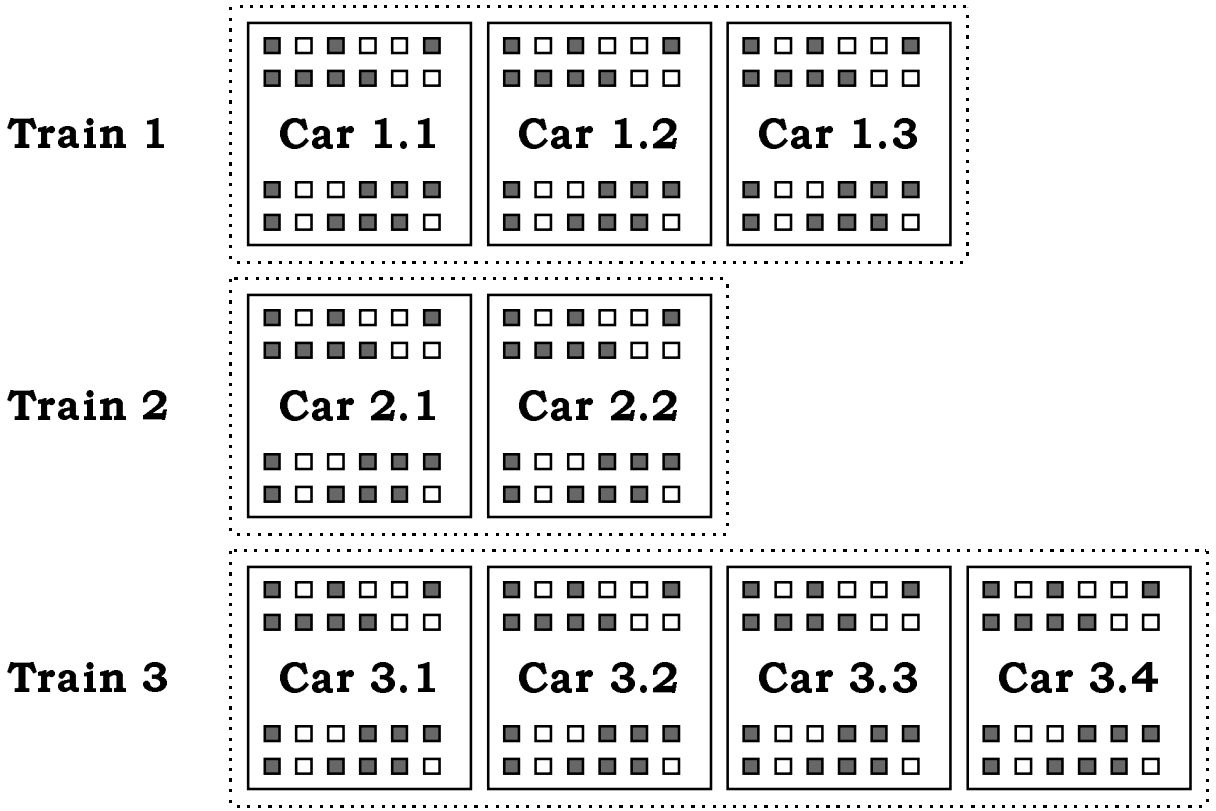


The Train Algorithm

- Provides *non-disruptive* collection of the oldest generation at a *low cost*
- Splits old generation into fixed-sized blocks (*cars*), organized into disjoint sets (*trains*)
- Processes one block at a time, in lexicographical order
- Recognizes and reclaims even very large cyclic garbage structures

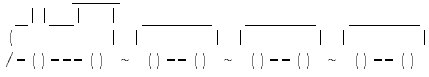


Old Generation Layout

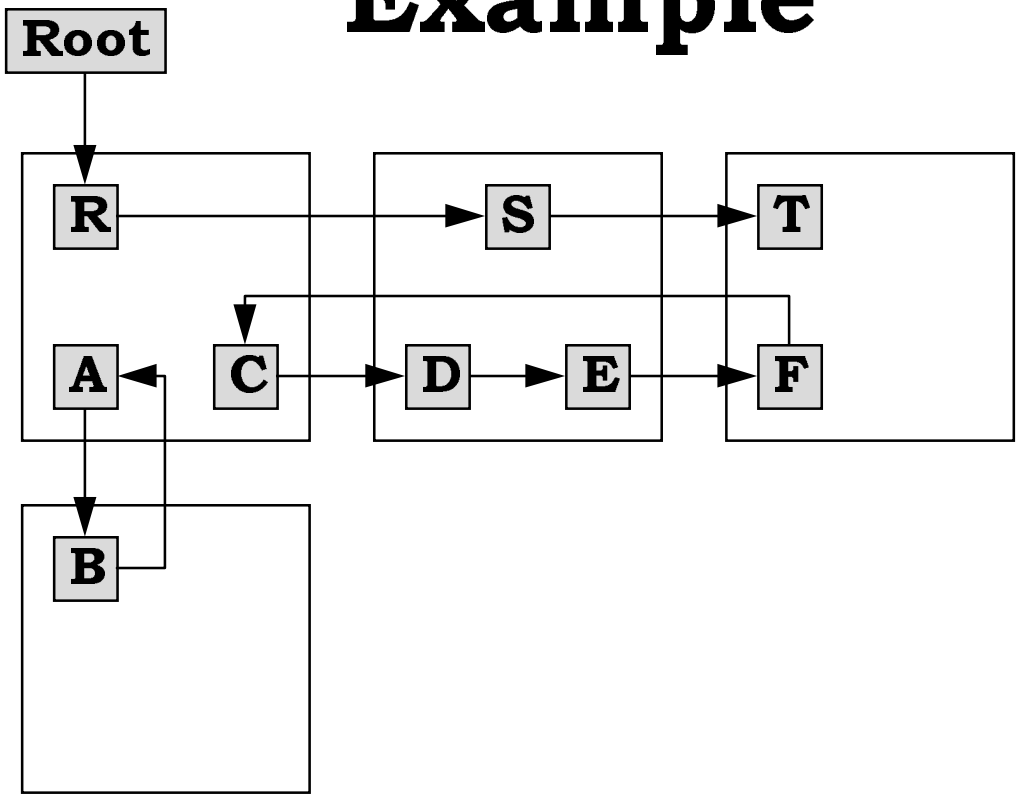


Car Collection Strategy

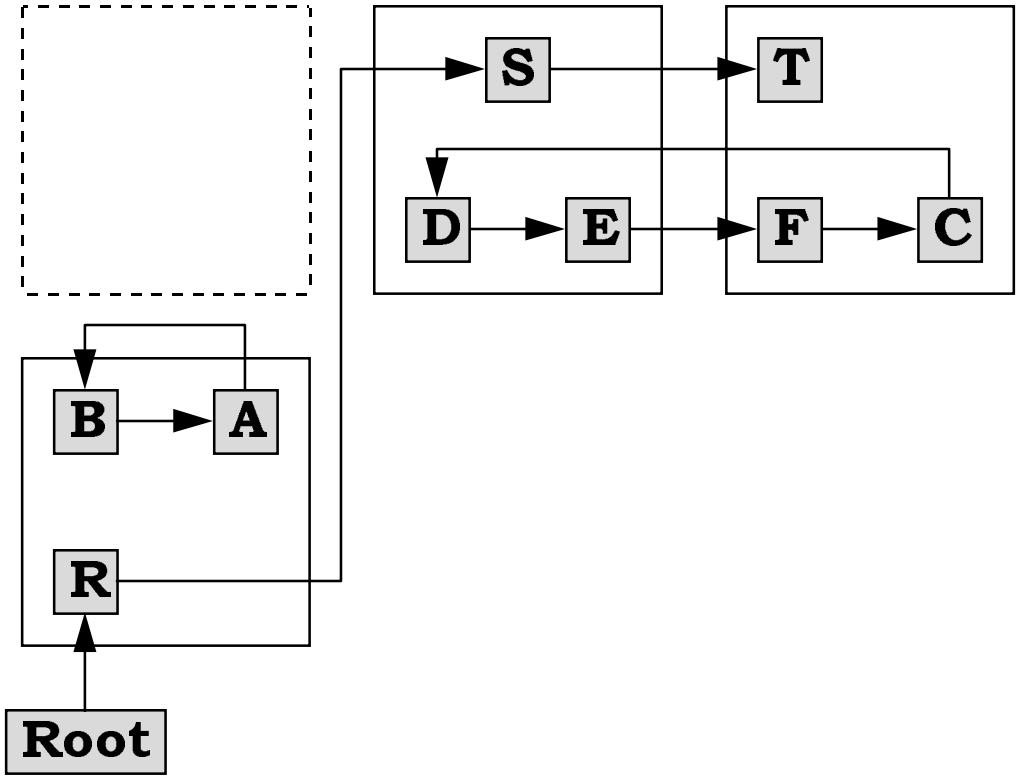
1. If there are no external references into the train being collected then reclaim all its cars
2. Recursively evacuate *externally* referenced objects to source trains
3. Recursively evacuate *internally* referenced objects to end of train
4. Reclaim car



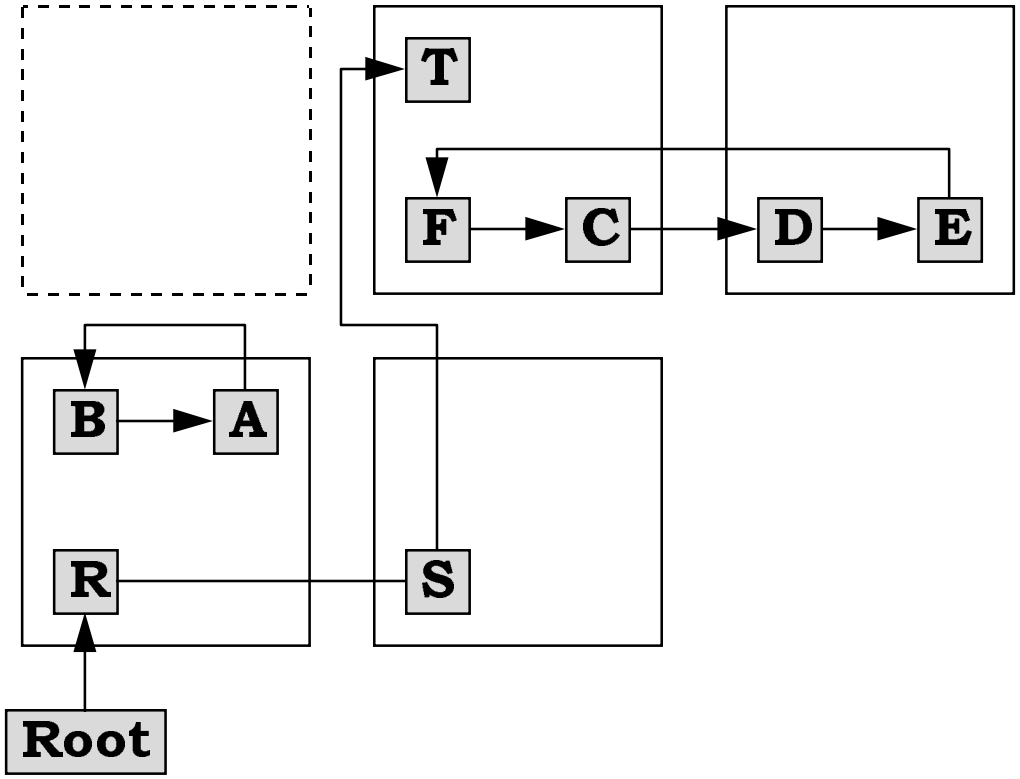
Example



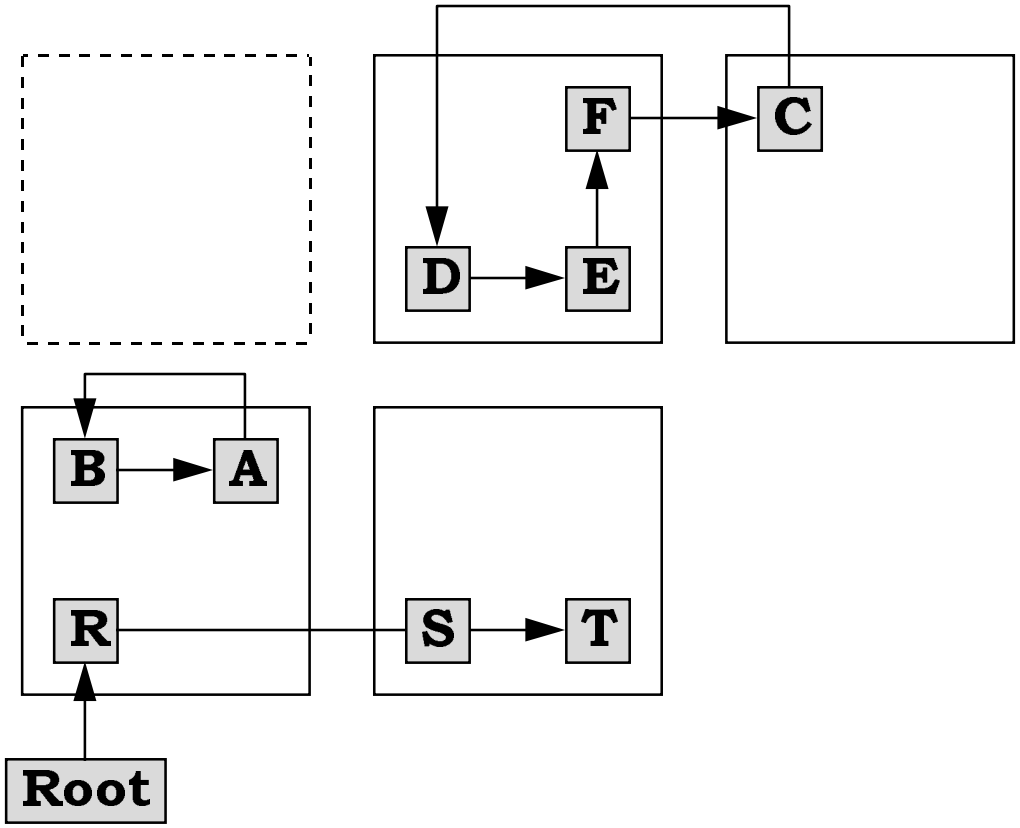
Example, continued



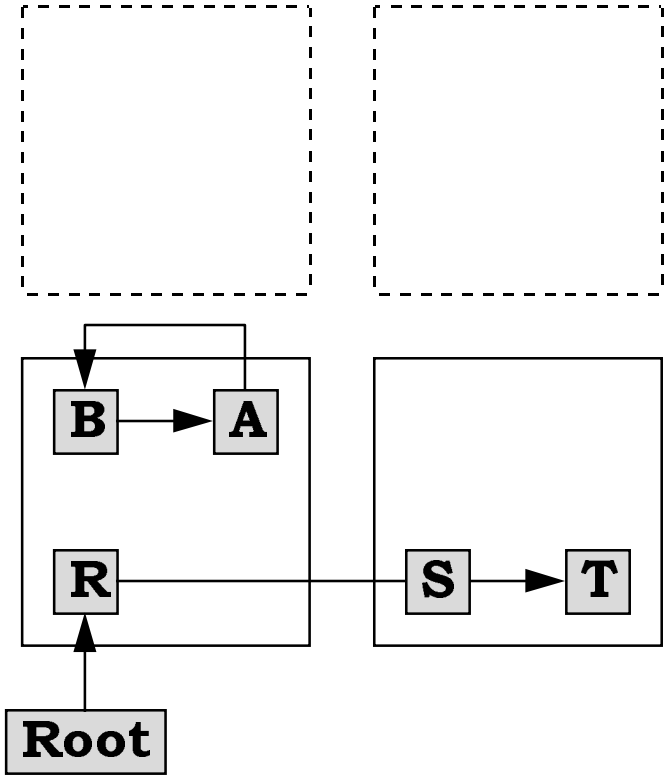
Example, continued



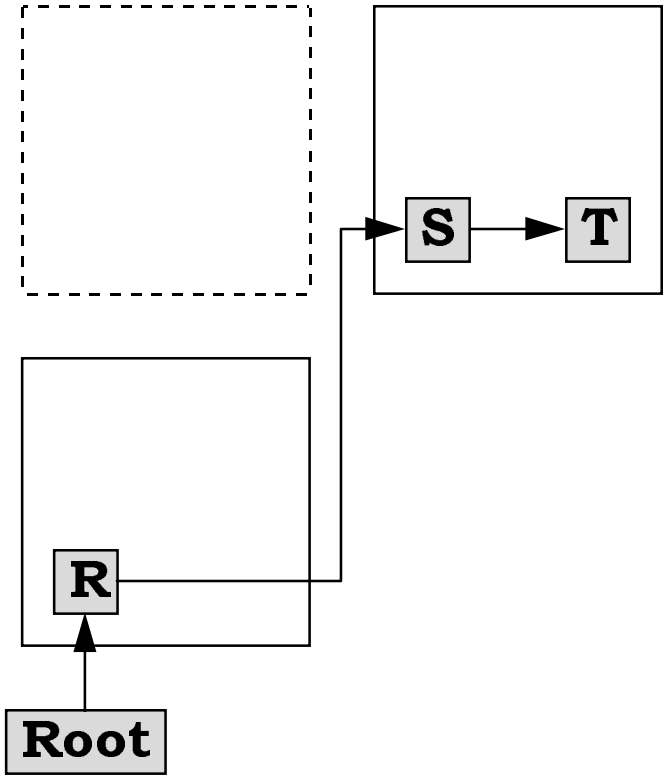
Example, continued



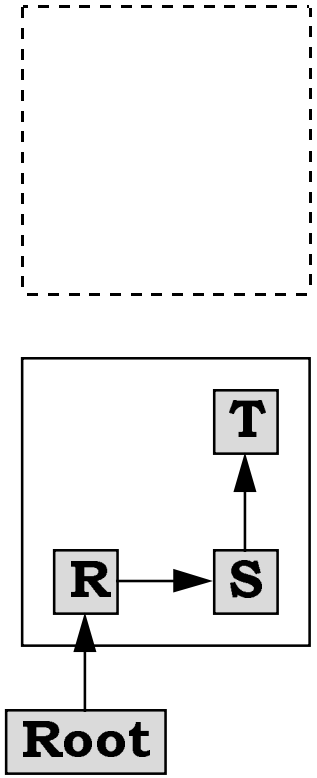
Example, continued



Example, continued



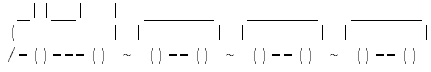
Example, continued



Technical Issues

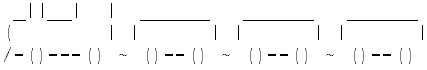
- **Remembered set for each car**
- **Extended run-time pointer assignment check (*write barrier*)**
- ***Train table* facilitates quick block look-up**

Address range	Train	Car
0 - 64 Kb	–	–
64 - 128 Kb	3	2
128 - 192 Kb	2	1
192 - 256 Kb	–	–
256 - 320 Kb	3	1
...	...	...



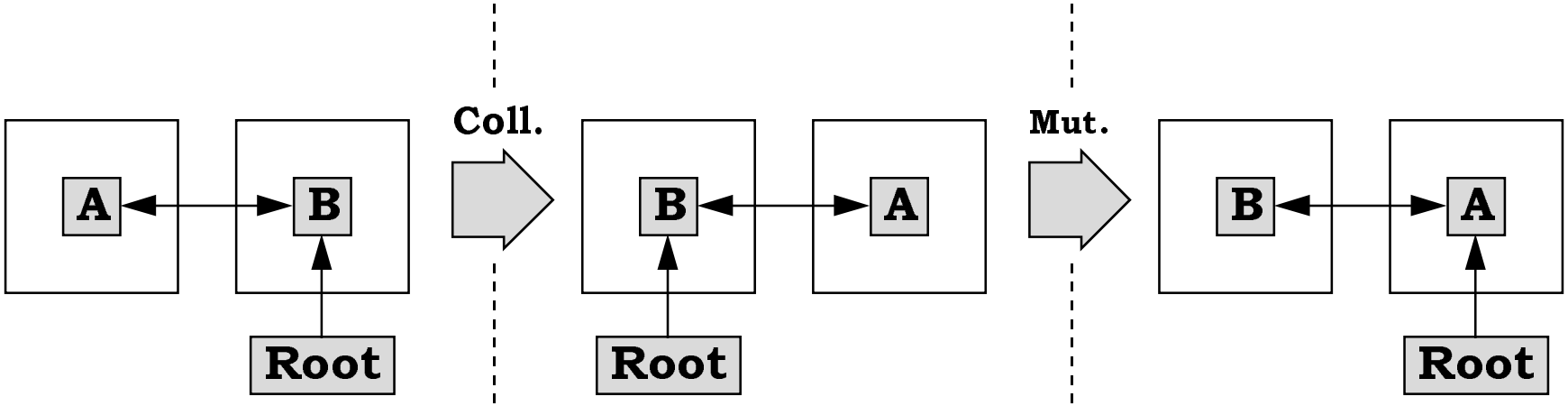
Correctness

- **The algorithm is *safe*, i.e. it never reclaims non-garbage**
- **The algorithm is *complete*, i.e. it reclaims all garbage**
  - **Garbage structures are collapsed into the highest-numbered train containing a member**
  - **All trains are processed**



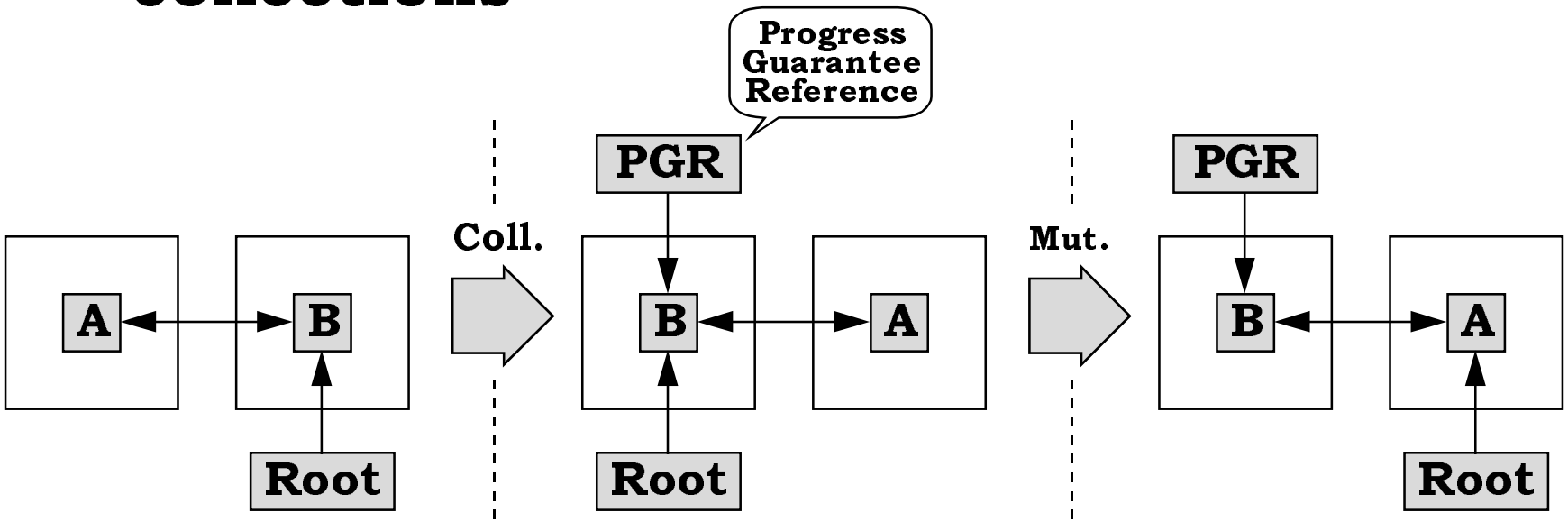
An Error

- **Continuos mutations of the object graph may prevent the algorithm from collecting any garbage!**



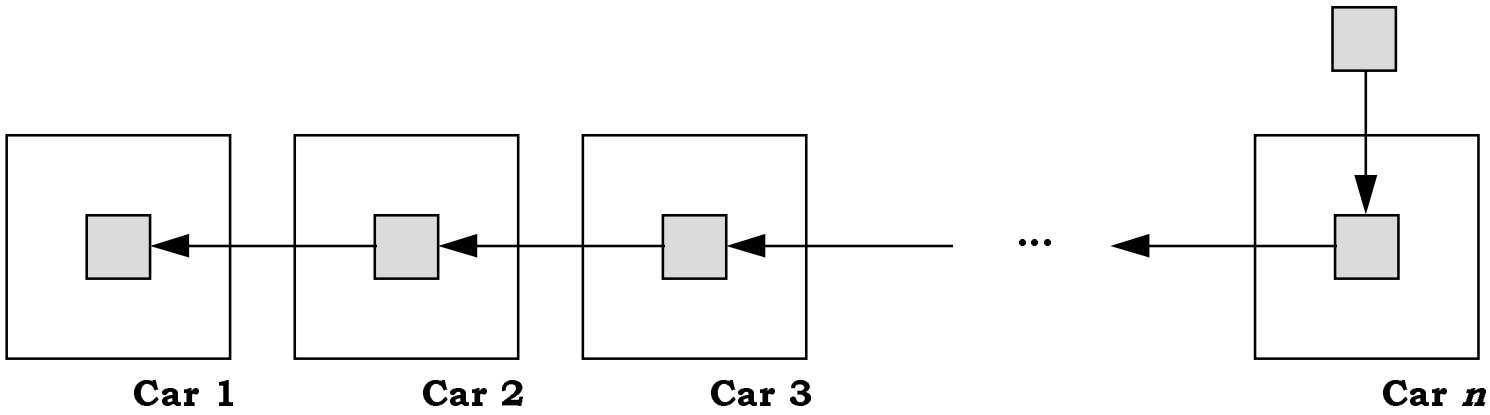
Correcting the Error

- **Progress may be ensured by explicitly recording an external pointer at *futile* collections**

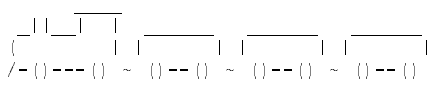


Complexity

- **Worst-case: O(n^2) collections may be required to reclaim a train of length *n***

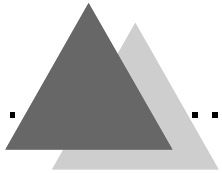


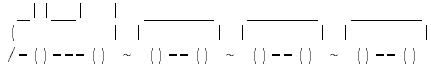
- **Observed: 1.13*n* (avg.), 2.00*n* (max.)**



Popular Objects

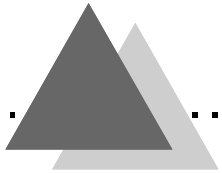
- Objects with many references to them are called *popular*
- It is desirable to avoid moving popular objects
- This can be done by moving cars containing popular objects *logically* rather than physically
- This is not as simple as it sounds!

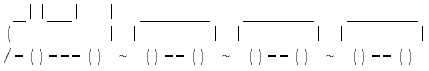




The Implementation

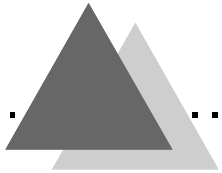
- **Run-time system:**
  - **Mjølner BETA System garbage collector**
  - **Two generations (copy collection + mark/sweep)**
- **Technical issues:**
  - **64 Kb car size**
  - **Adaptive invocation frequency scheme**
  - **Special treatment of (highly referenced) basic environment object**

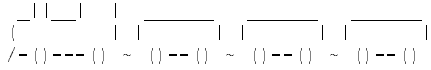




The Results

- **Benchmark system:**
  - **SUN SPARC IPX 4/50, 40 MHz, 32 Mb**
- **Benchmark programs:**
  - **Mjølner BETA compiler compilation**
  - **Sif hyper-structure editor session**
  - **Discrete event simulation**





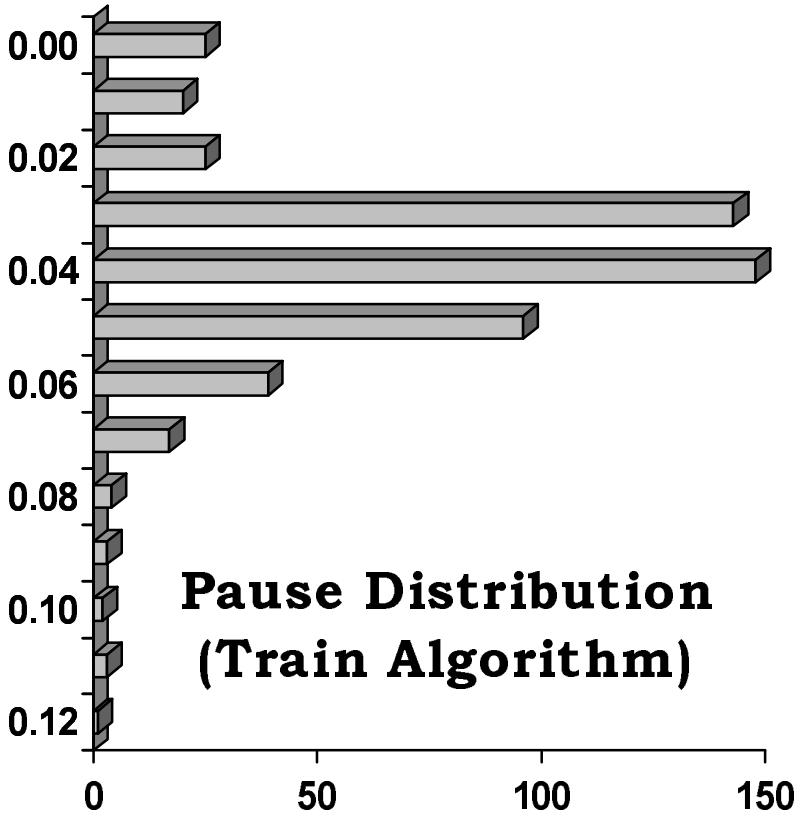
Collection Pauses

Method	Batch	Train
Coll.s	10	528

Collections

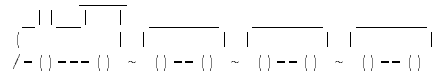
Method	Batch	Train
Min.	0.27s	0.00s
Avg.	1.71s	0.04s
Max.	3.21s	0.12s

Collection Pauses



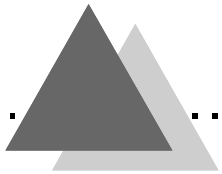
Run-time Overhead

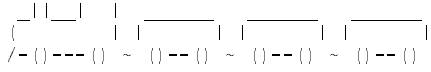
Method	Batch	Train	
Old	17s	21s	+20%
Young + Mutator	956s	959s	+0.3%
Total	973s	980s	+0.6%



Storage Overhead

Method	Batch	Train	
Objects	12.5 Mb	12.5 Mb	
Misc. bookkeeping	0.5 Mb	0.5 Mb	
Misc. train bookkeeping (inter-car remembered sets, train table)	–	1 Mb	
Unprocessed garbage	0.5 Mb	1 Mb	
Total	13.5 Mb	15.0 Mb	+10%



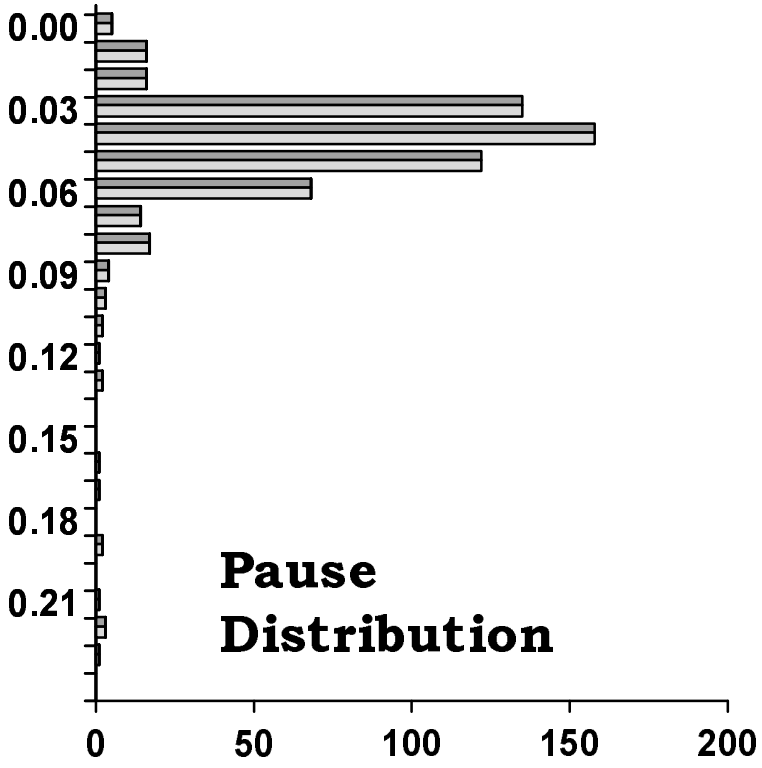


Popular Object Scheme

- Over 50,000 references to basic environment object

Popular	On	Off
Min.	0.00s	0.00s
Avg.	0.04s	0.05s
Max.	0.12s	0.23s

Collection Pauses

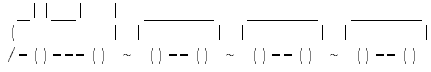


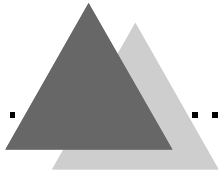
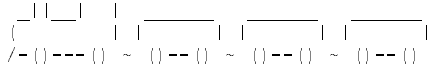
Virtual Memory Behavior

- **Benchmark system:**
  - **SUN SPARC ELC 4/25, 33 MHz, 12 Mb**

Method	Batch	Train
Collections	10	528
Page faults	3778	4476
PF/coll. (avg.)	378	8.5
PF/coll. (max.)	>1000	<100

+18%



References

- **Hudson & Moss: "Incremental Collection of Mature Objects", IWMM 1992**
  `ftp://ftp.cs.umass.edu/pub/osl/papers/iwmm92.ps.Z`
- **Grarup & Seligmann: "Incremental Mature Garbage Collection", M.Sc. Thesis, Aarhus University, 1993**
  `ftp://ftp.daimi.aau.dk/pub/thesis/gcthesis.ps.Z`
- **Seligmann & Grarup: "Incremental Mature Garbage Collection Using the Train Algorithm", ECOOP 1995**
  `http://www.daimi.aau.dk/~beta/Papers/Train/`