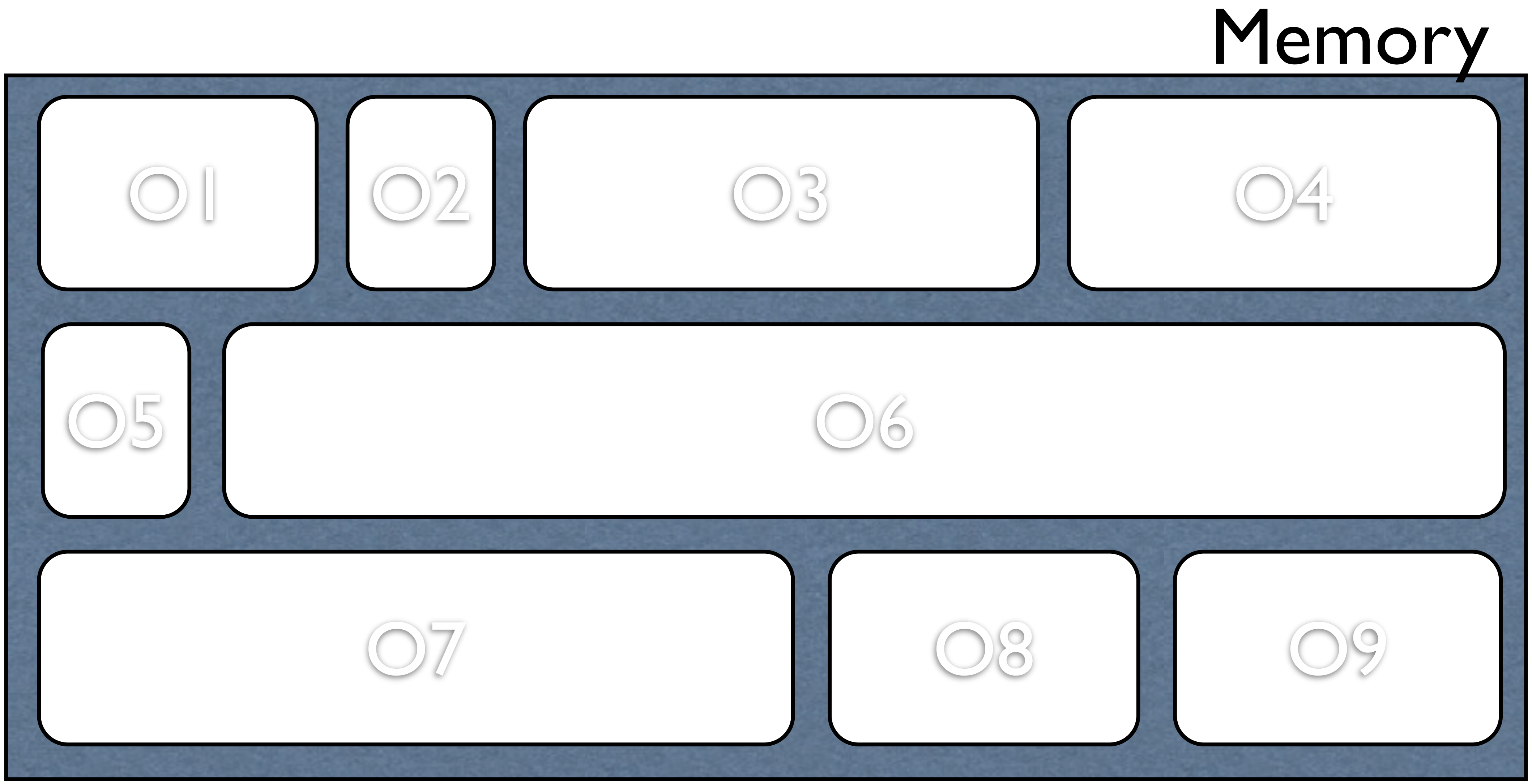
Comparison of Compacting Algorithms for Garbage Collection

Ahmed Hussein

Agenda

- Compaction..What is that?
- Presenting four different algorithms
  - Lisp2
  - Table Compactors
  - Morris
  - Jonkers

Overview



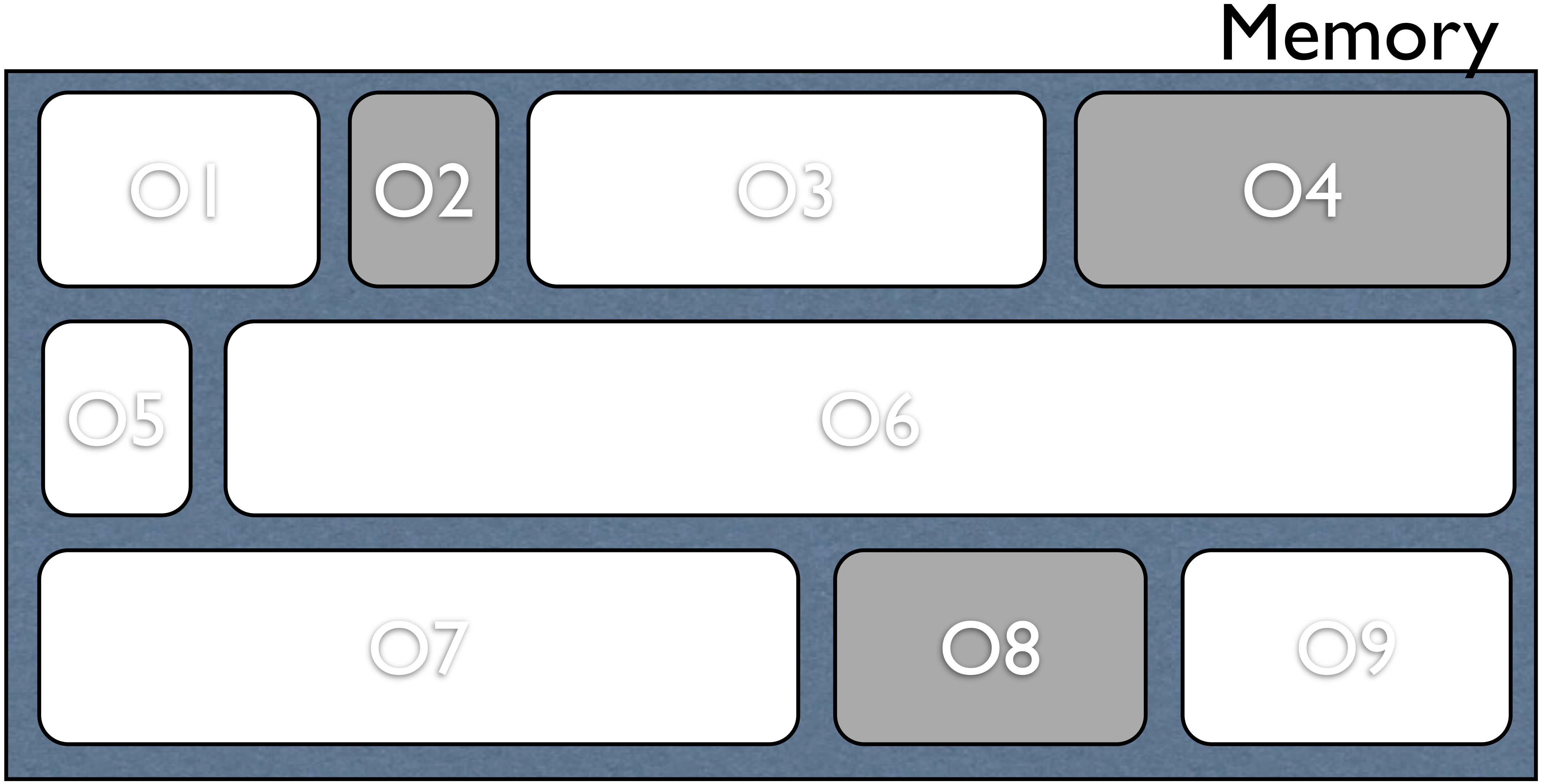
Memory

O1 O2 O3 O4

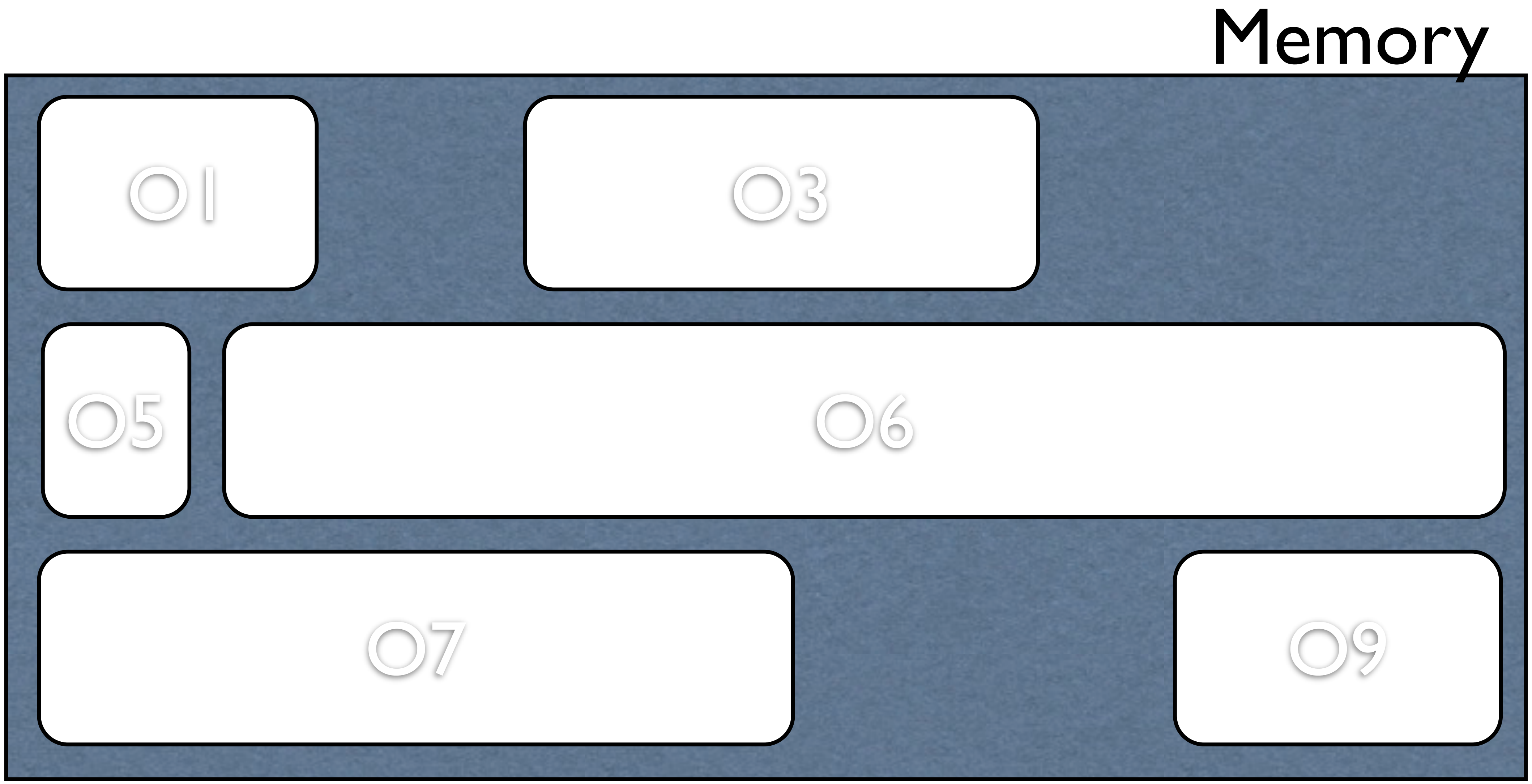
O5 O6

O7 O8 O9

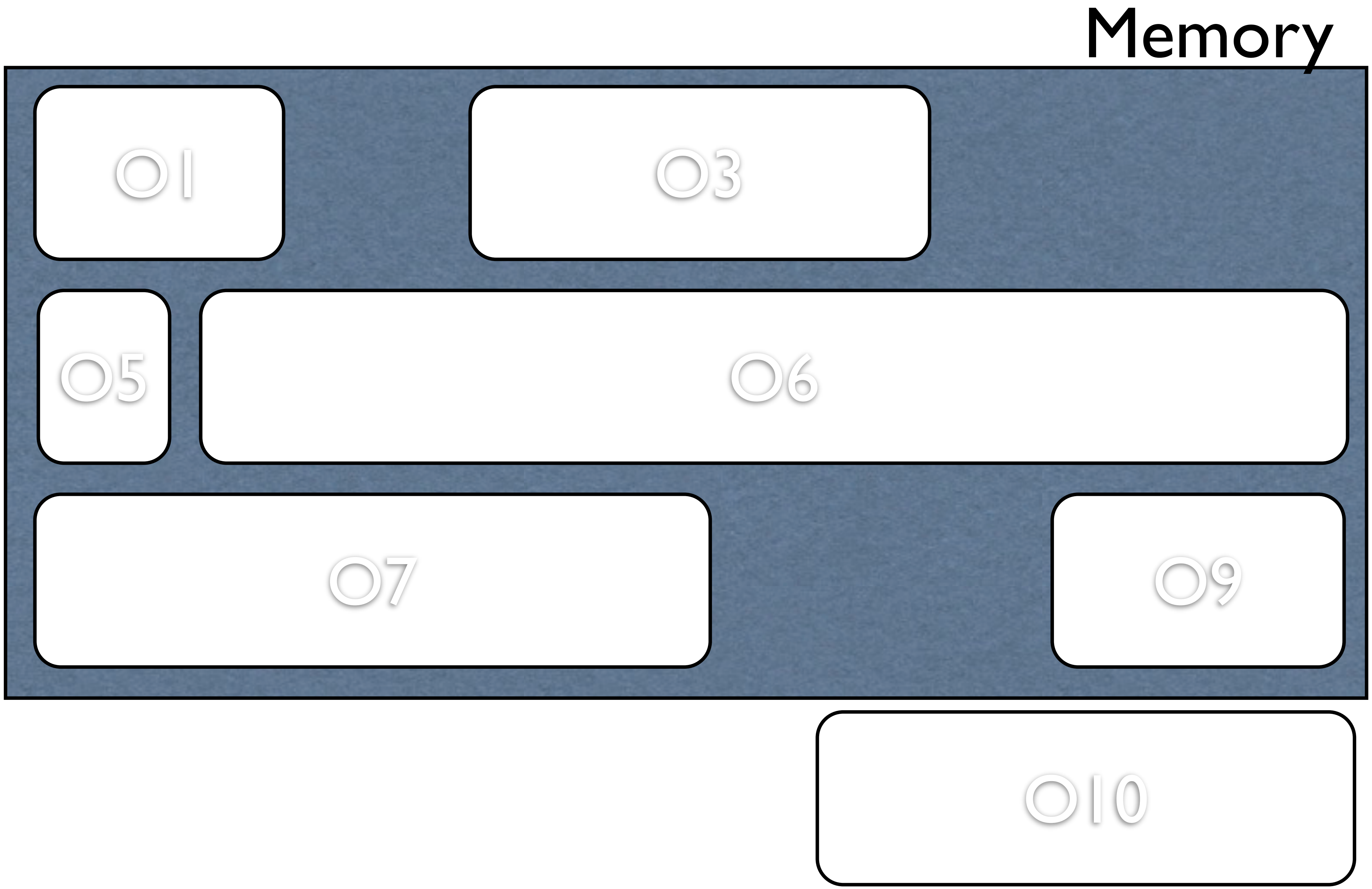
Phase1..Marking



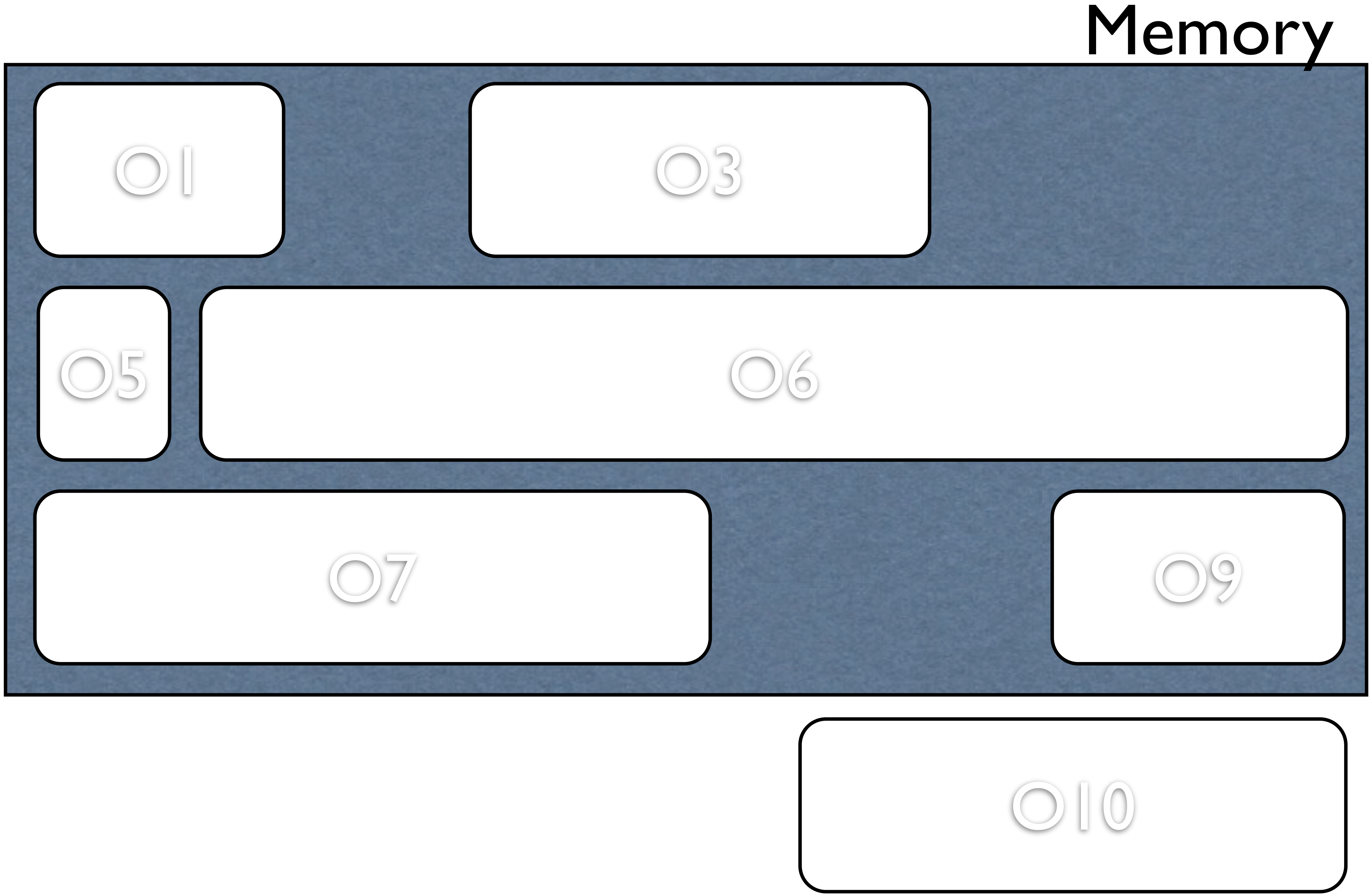
Phase2..Collecting



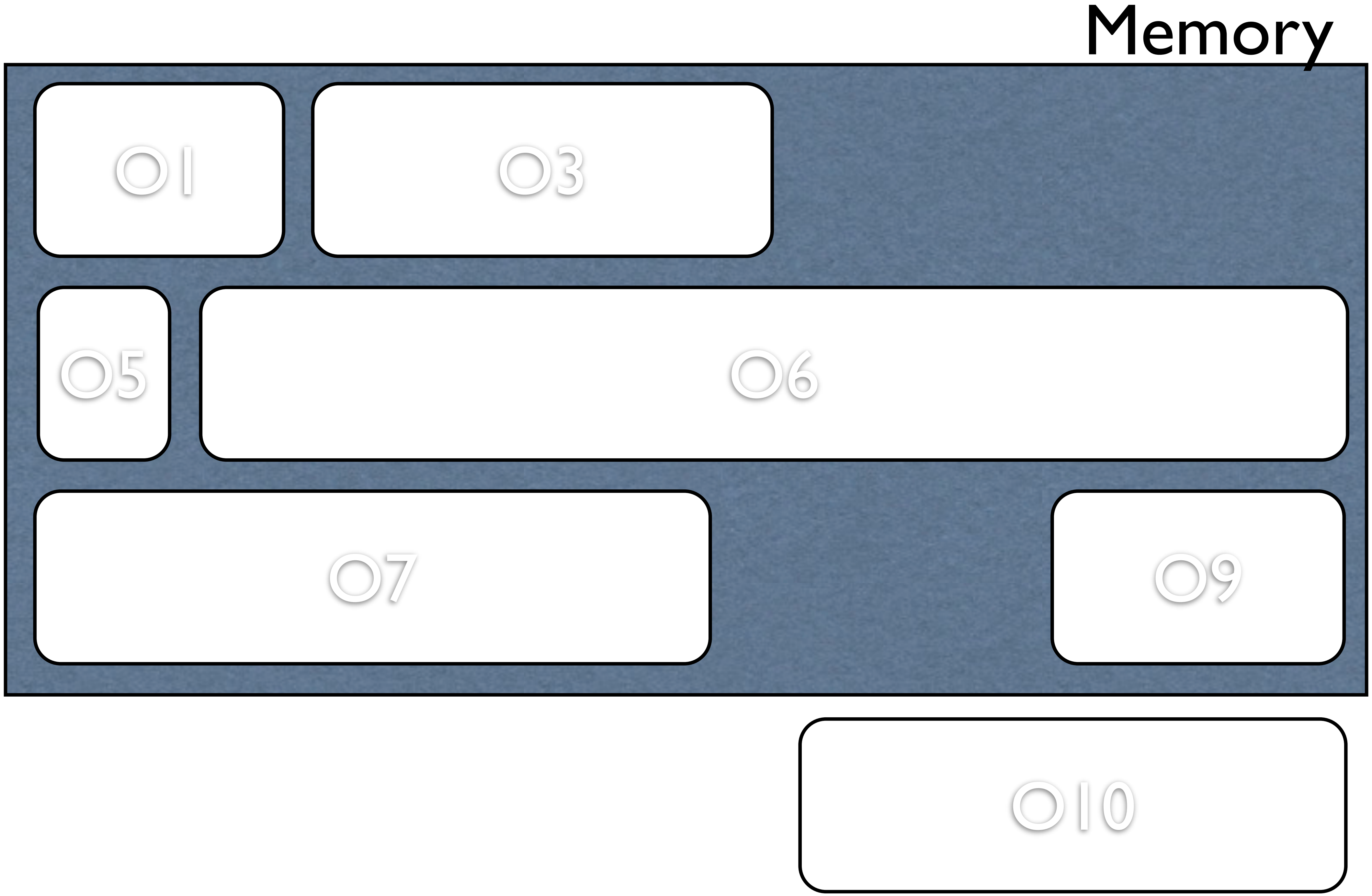
Phase2..Collecting



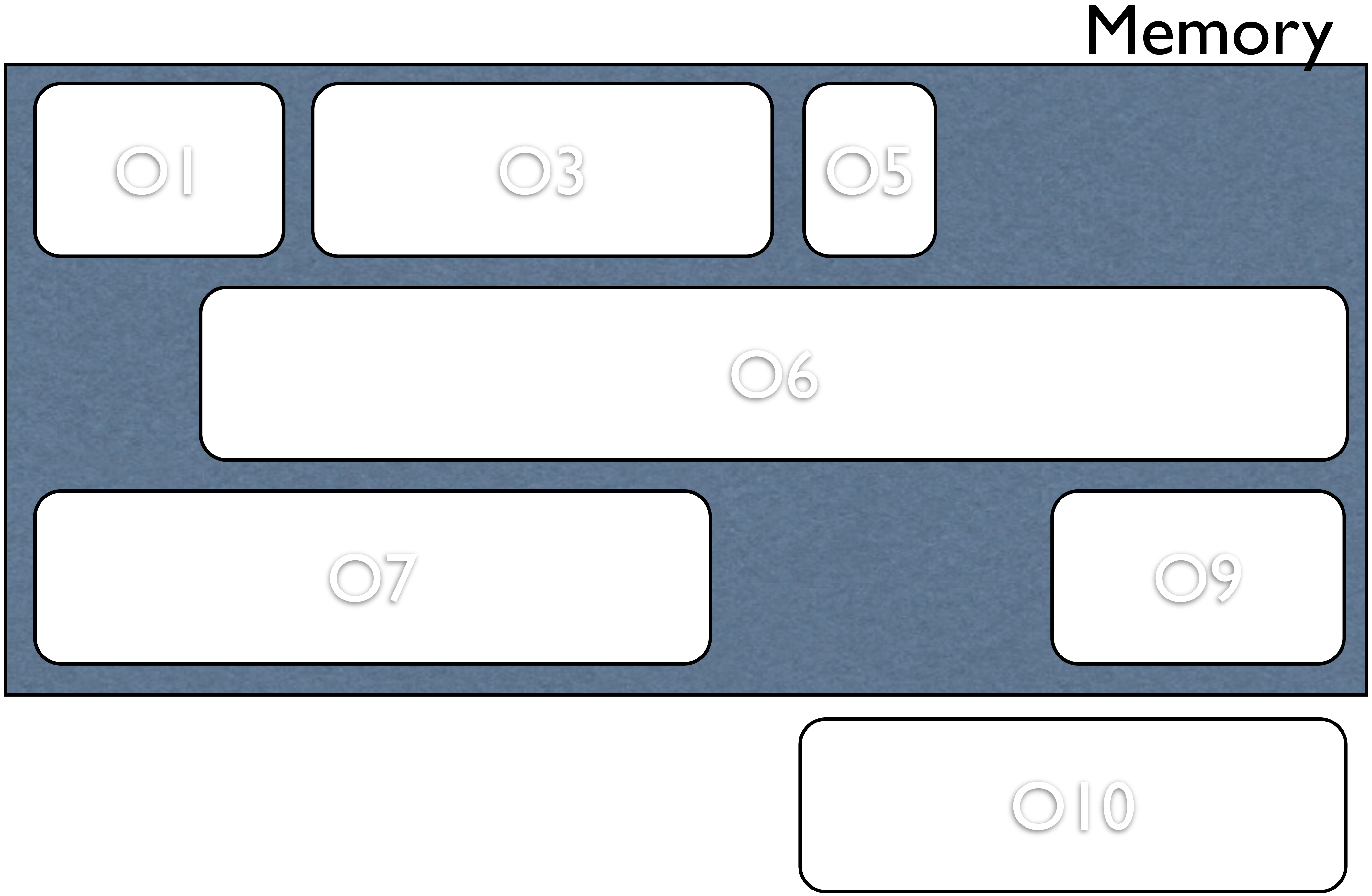
Phase3..Compaction



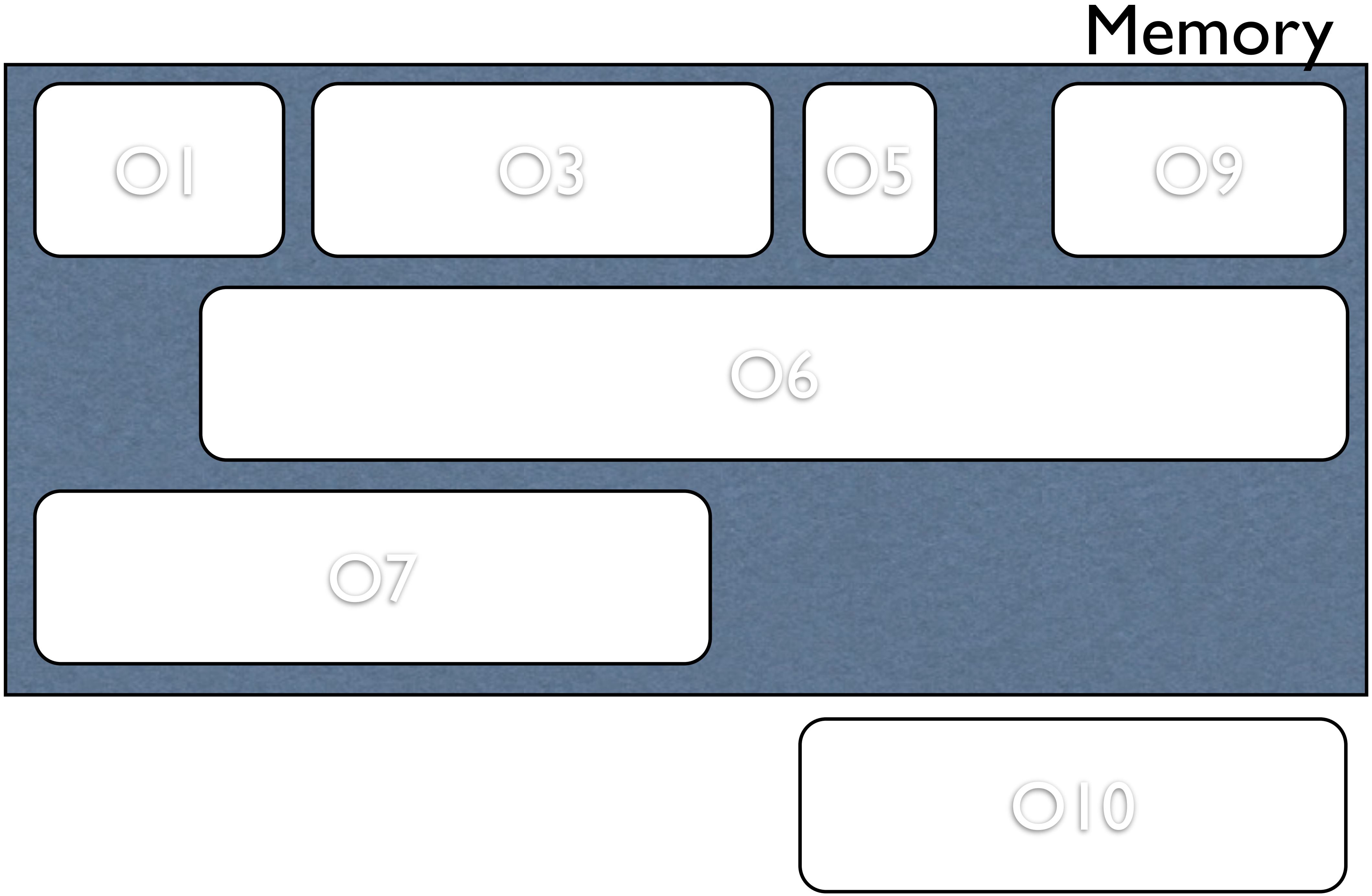
Phase3..Compaction



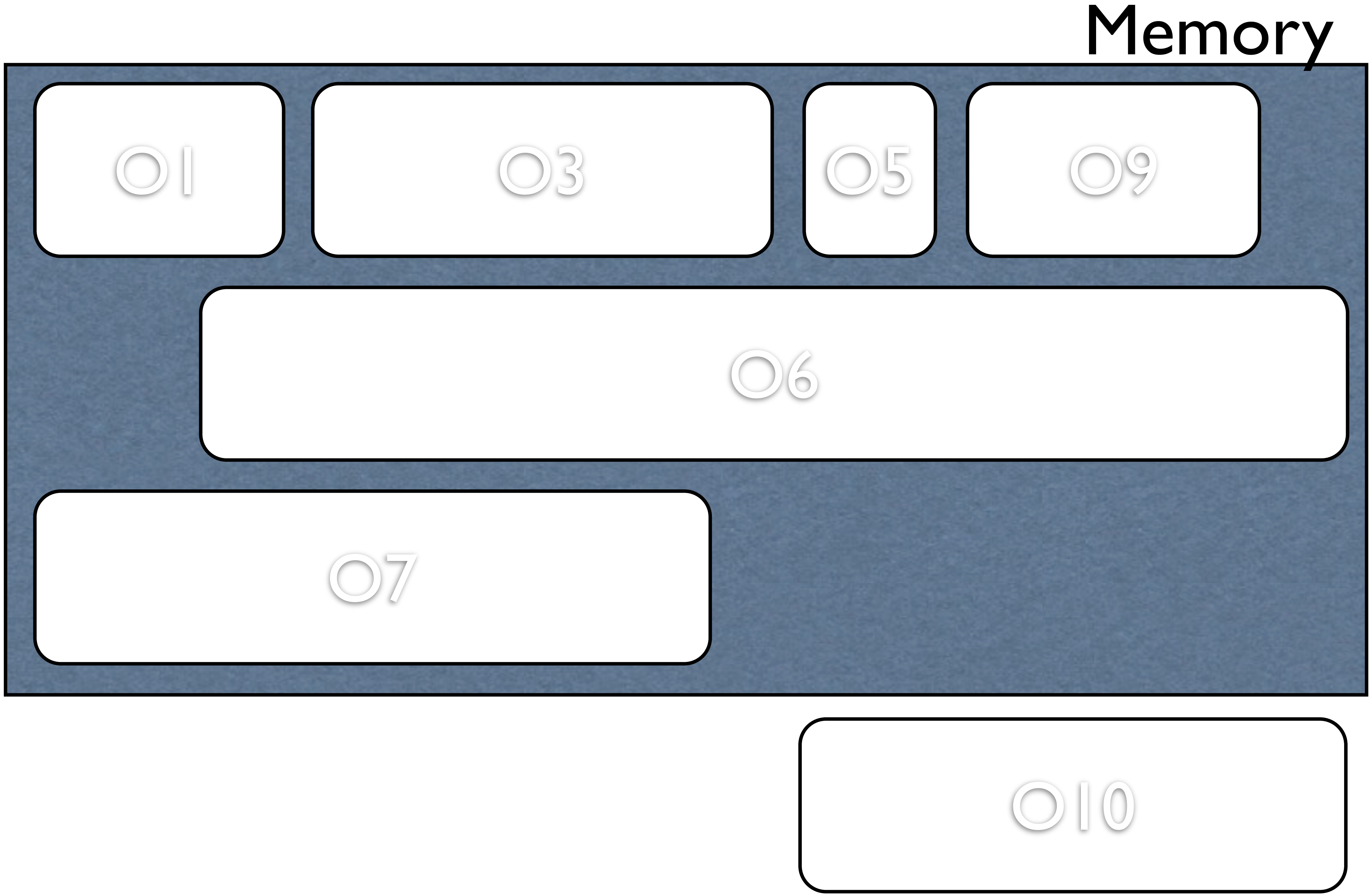
Phase3..Compaction



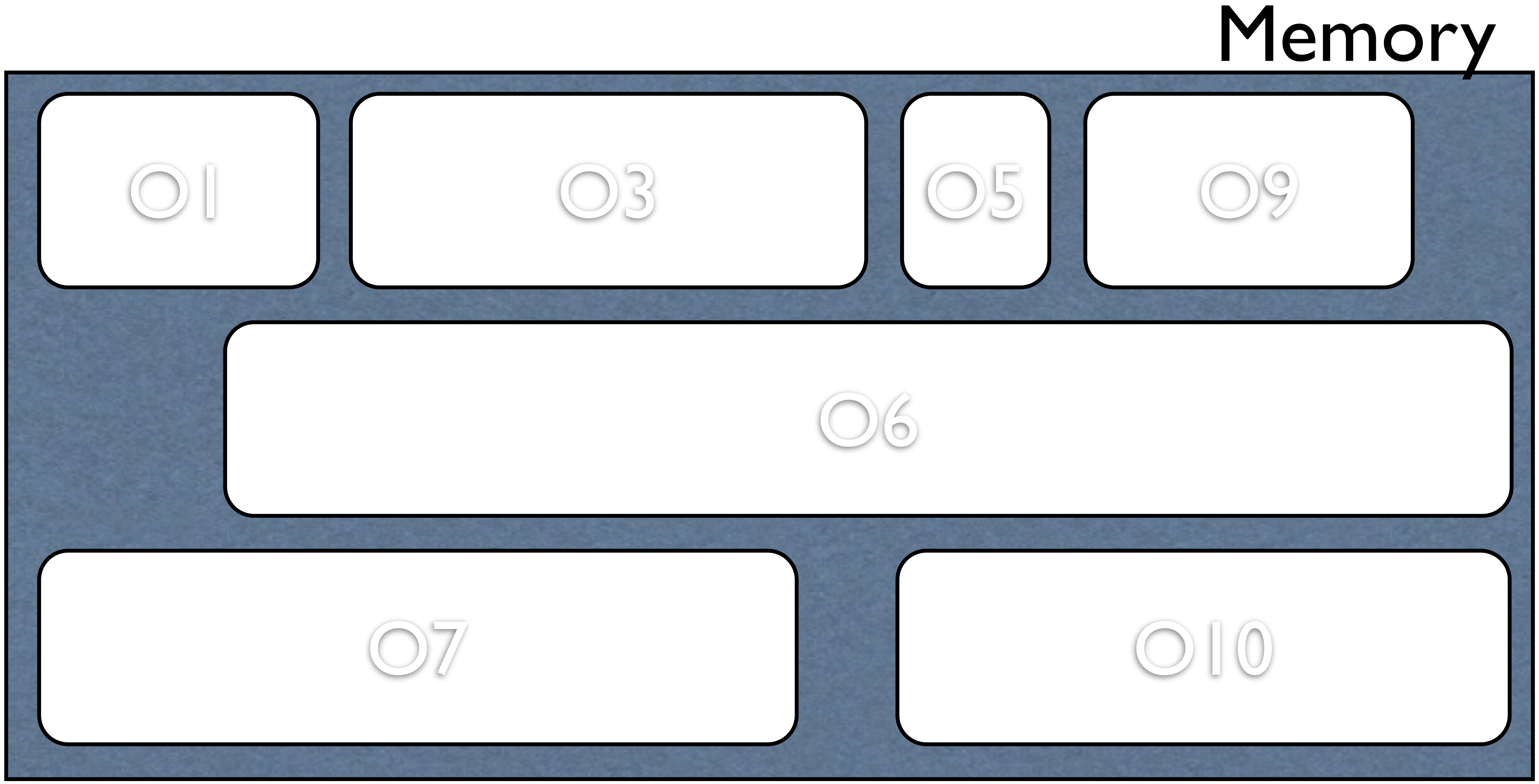
Phase3..Compaction



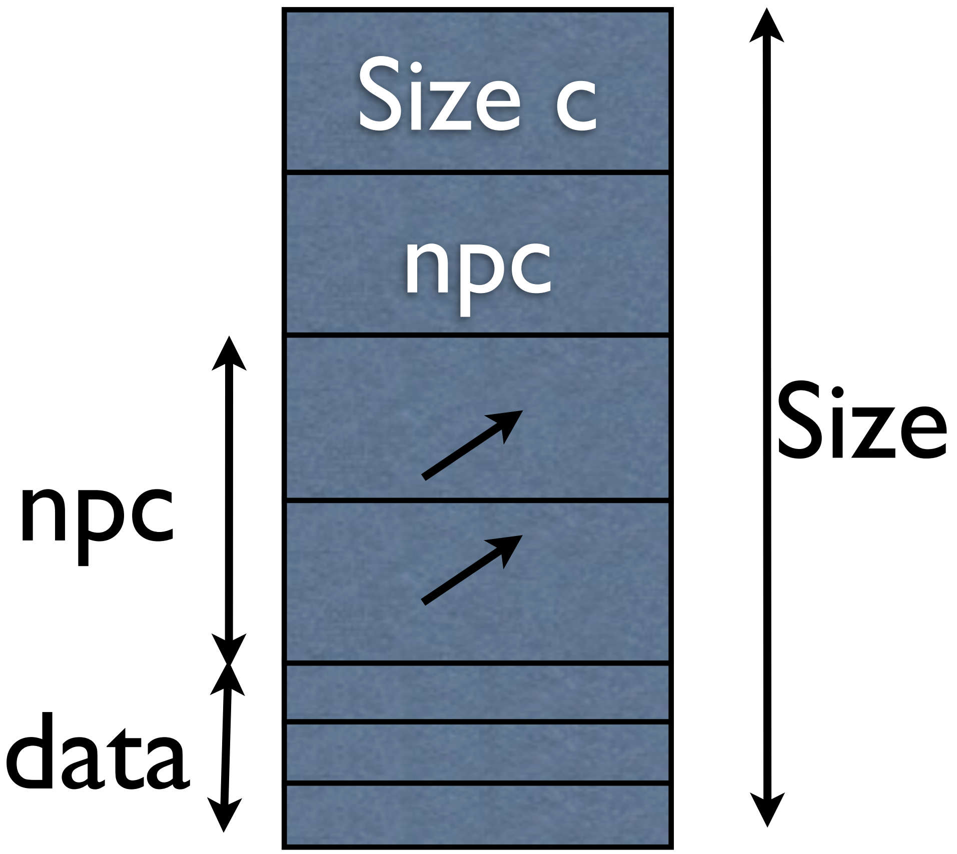
Phase3..Compaction



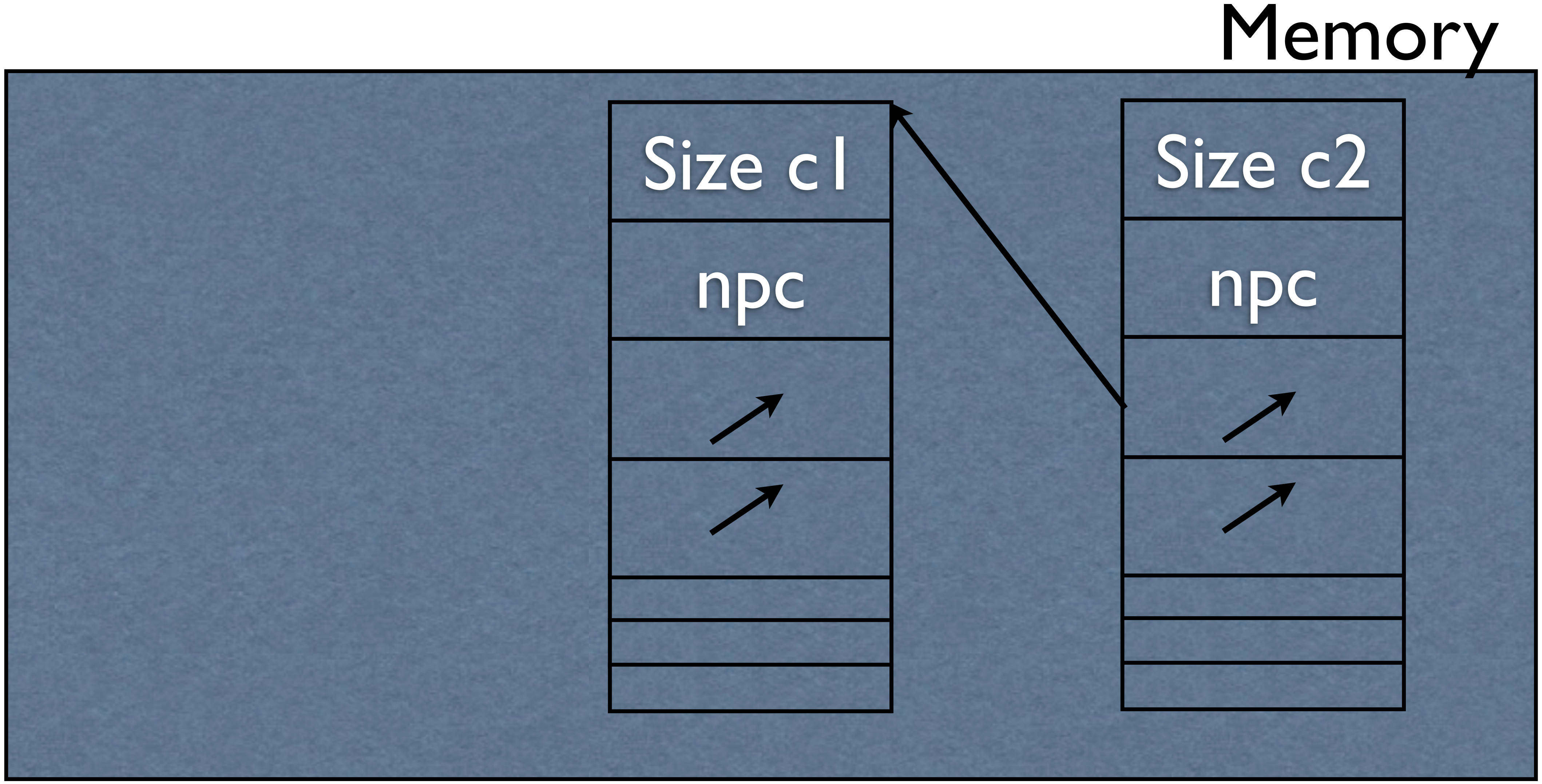
Phase3..Compaction



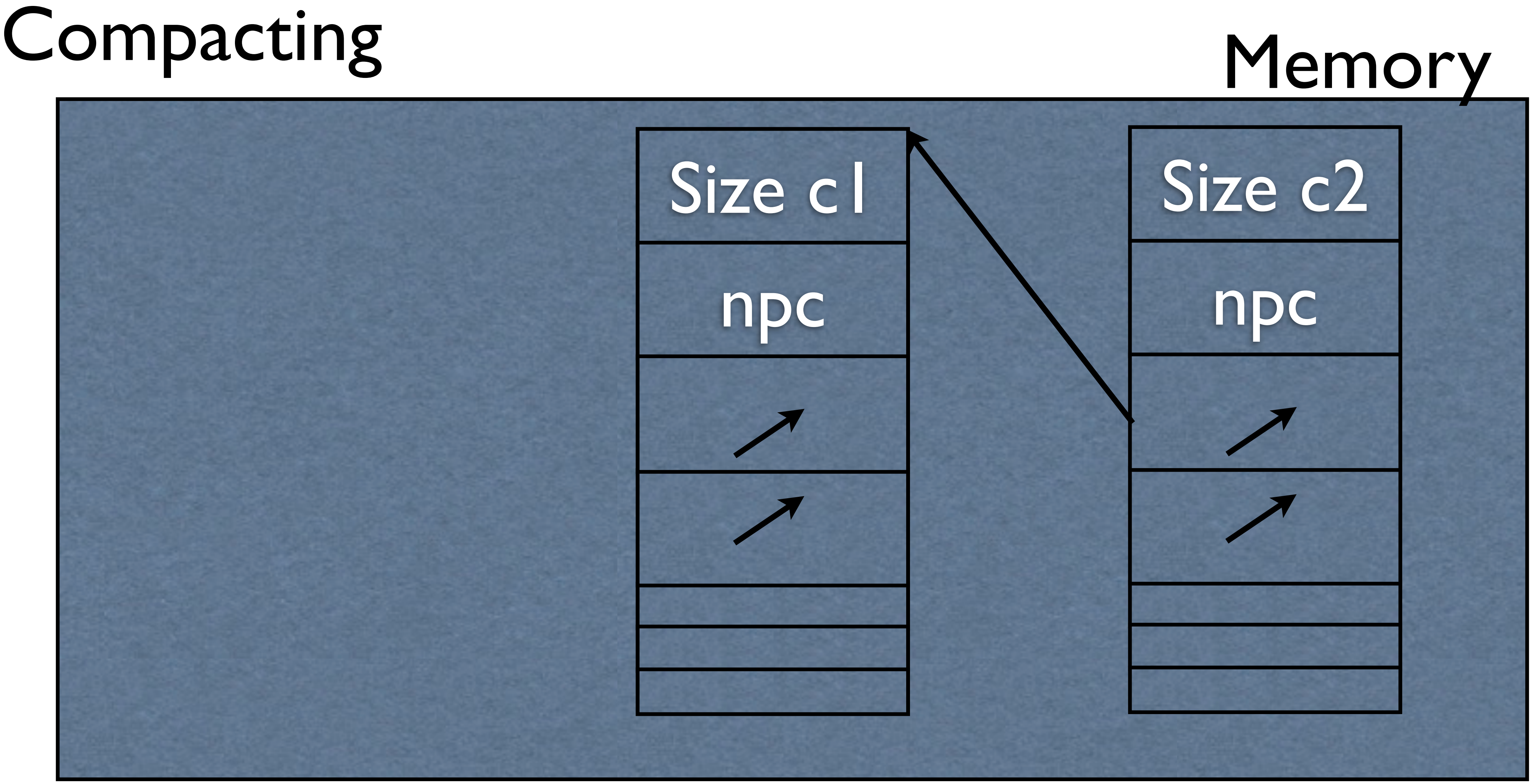
Object Model



Lisp2

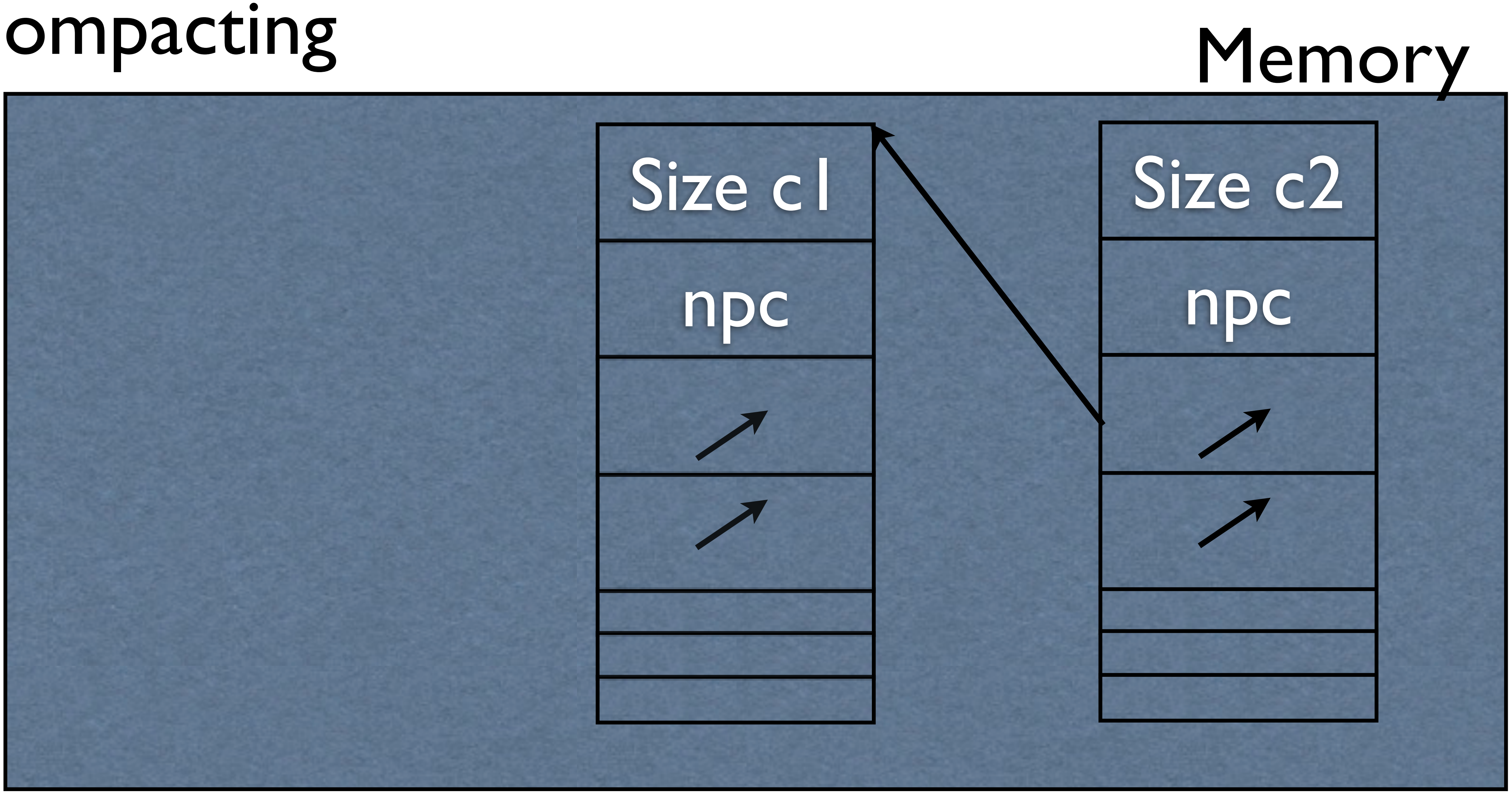


Lisp2

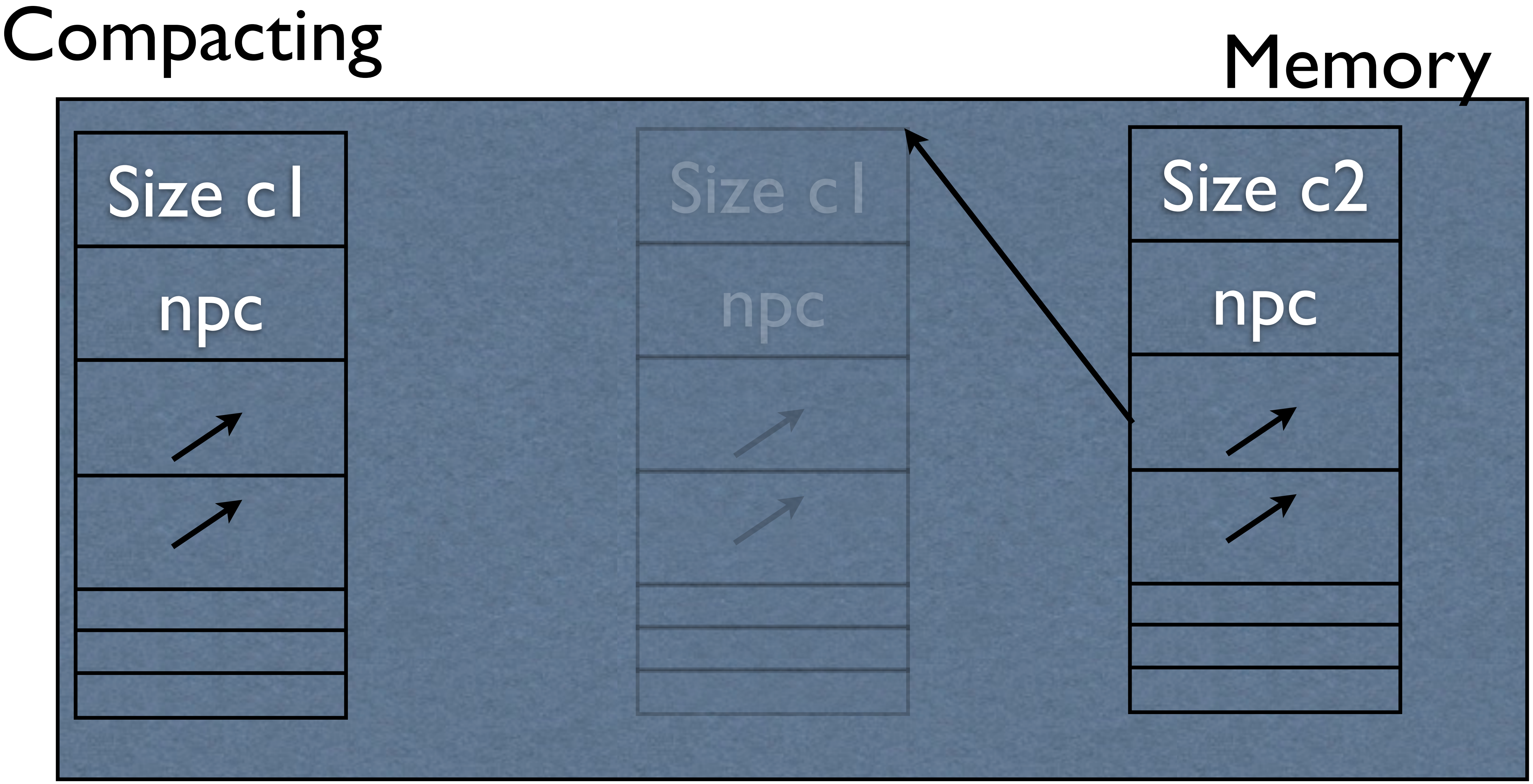


Lisp2

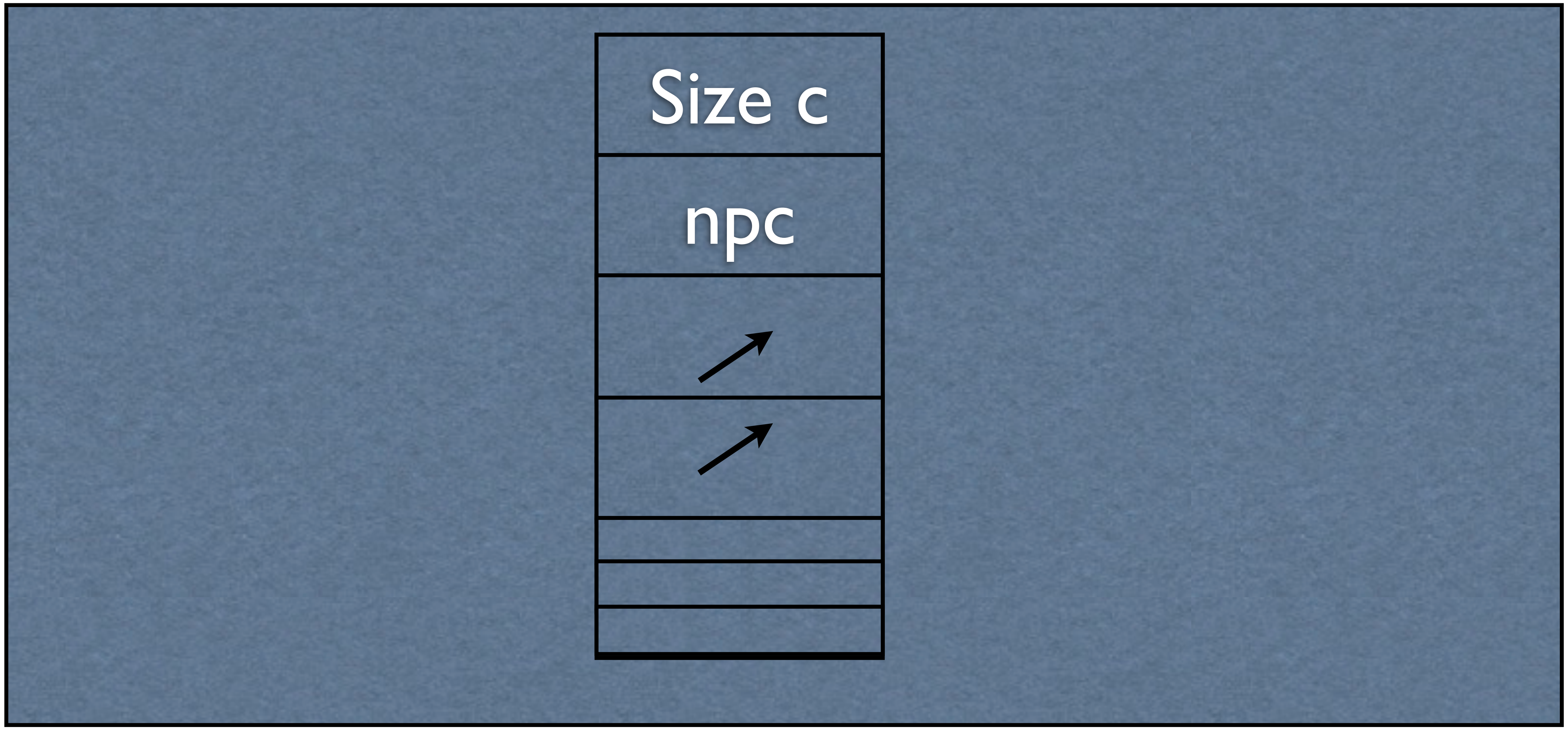
Compacting



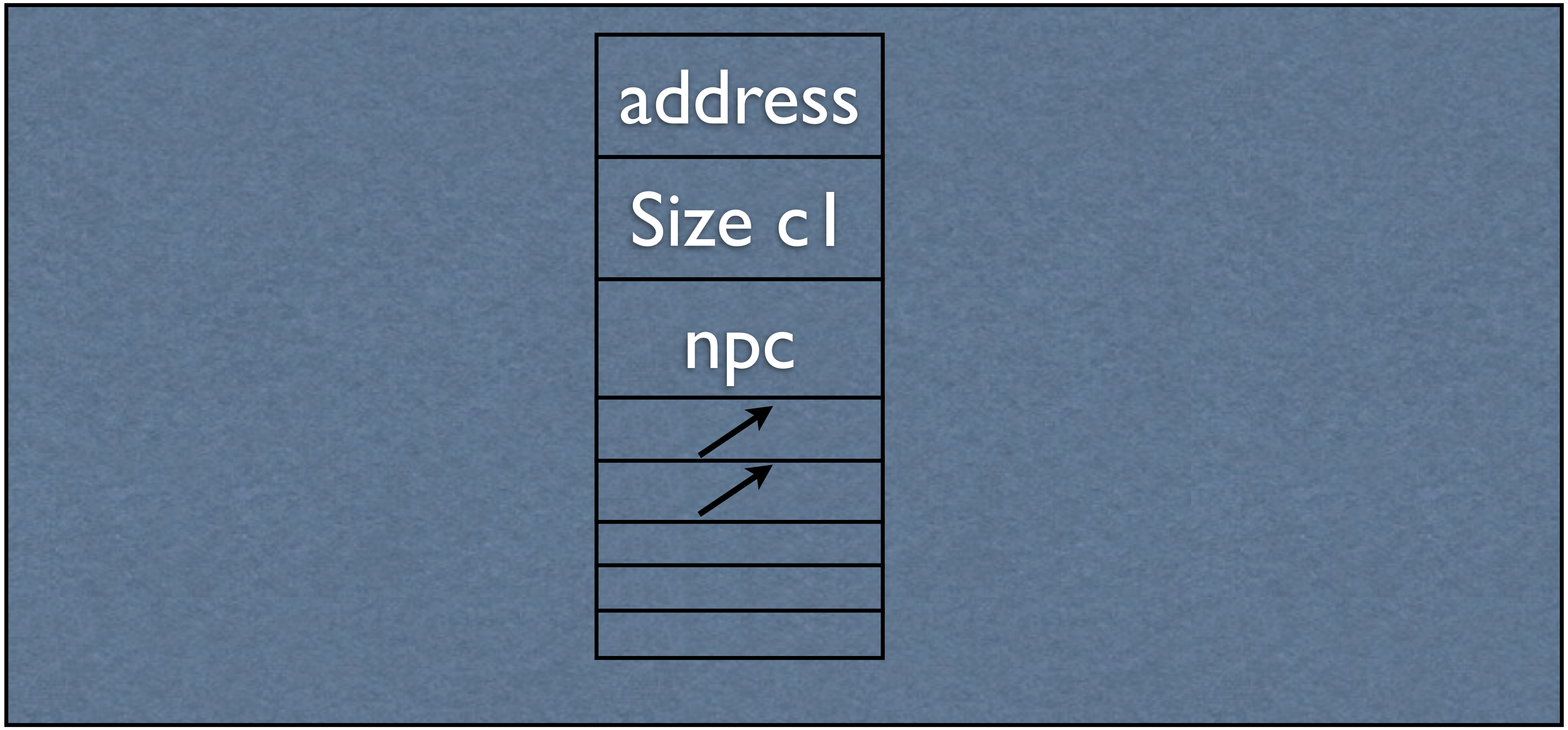
Lisp2



Lisp2

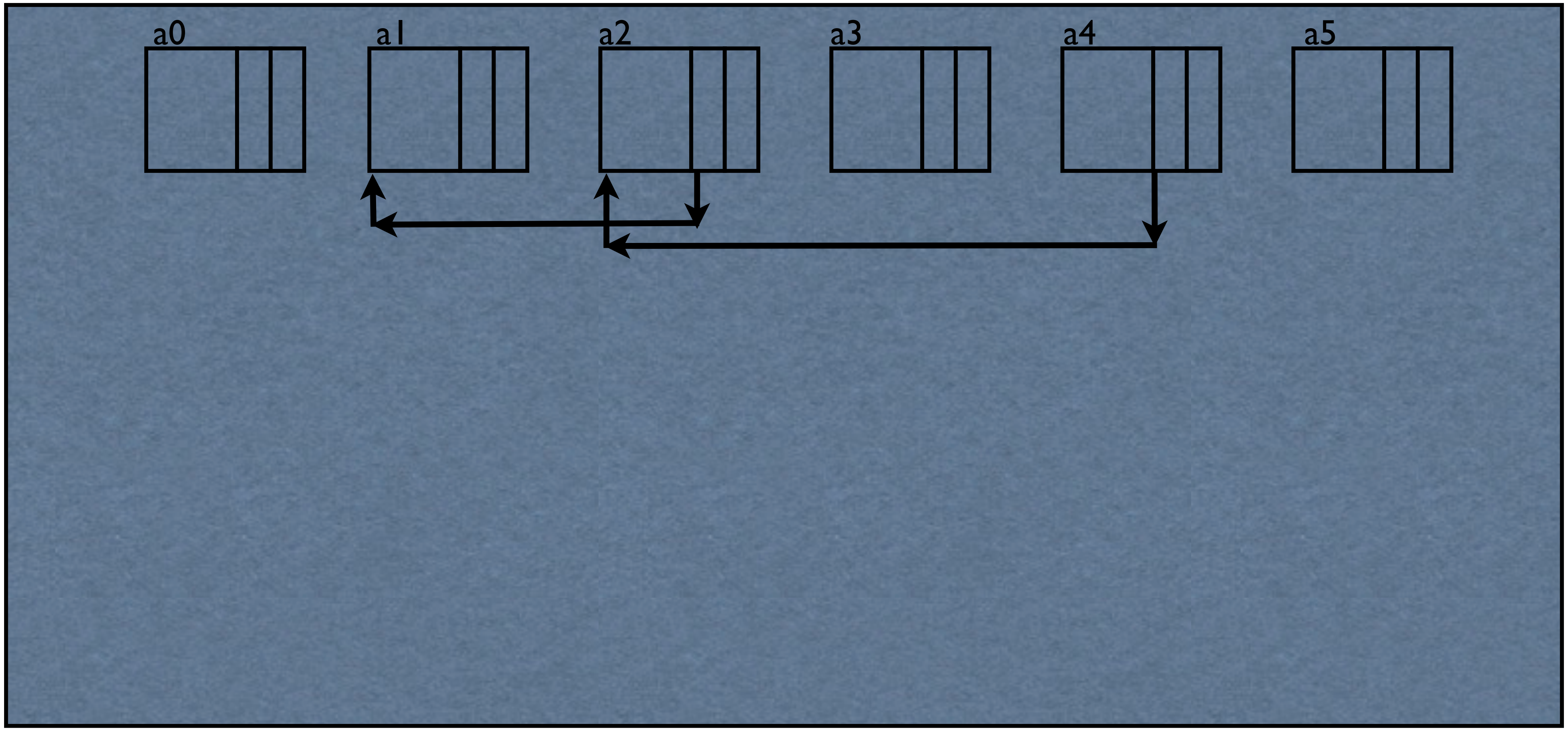


Lisp2



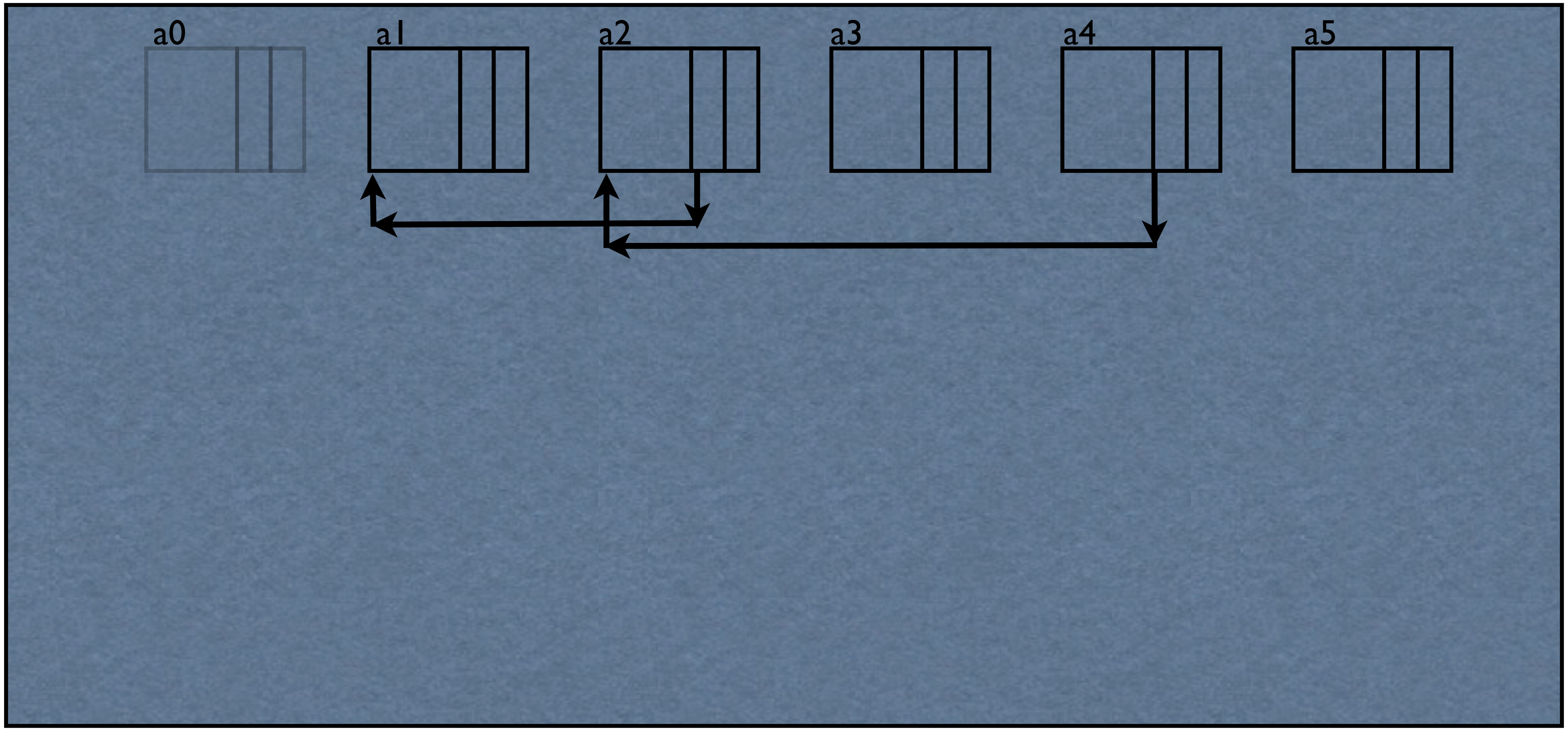
Lisp2

Pass1



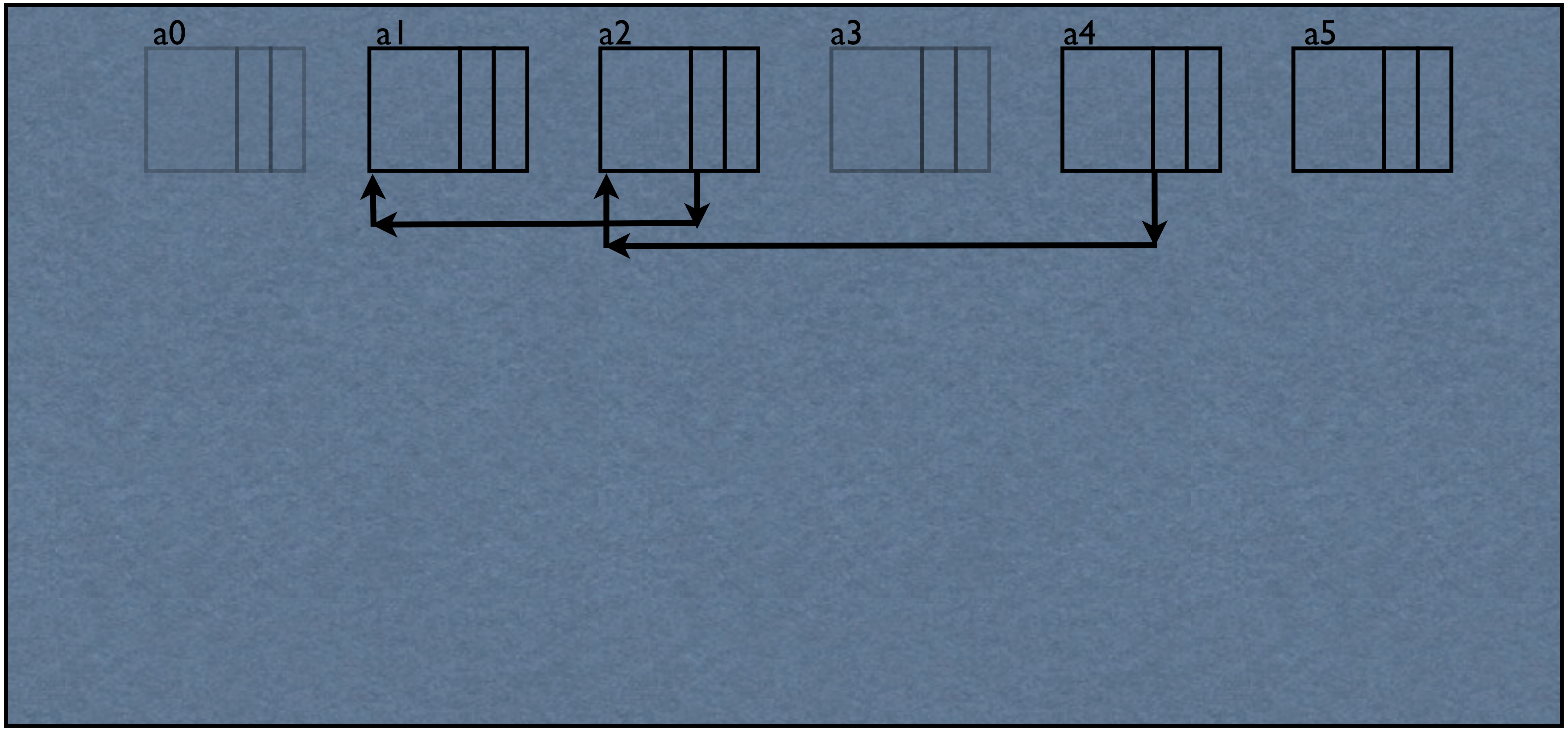
Lisp2

Pass1



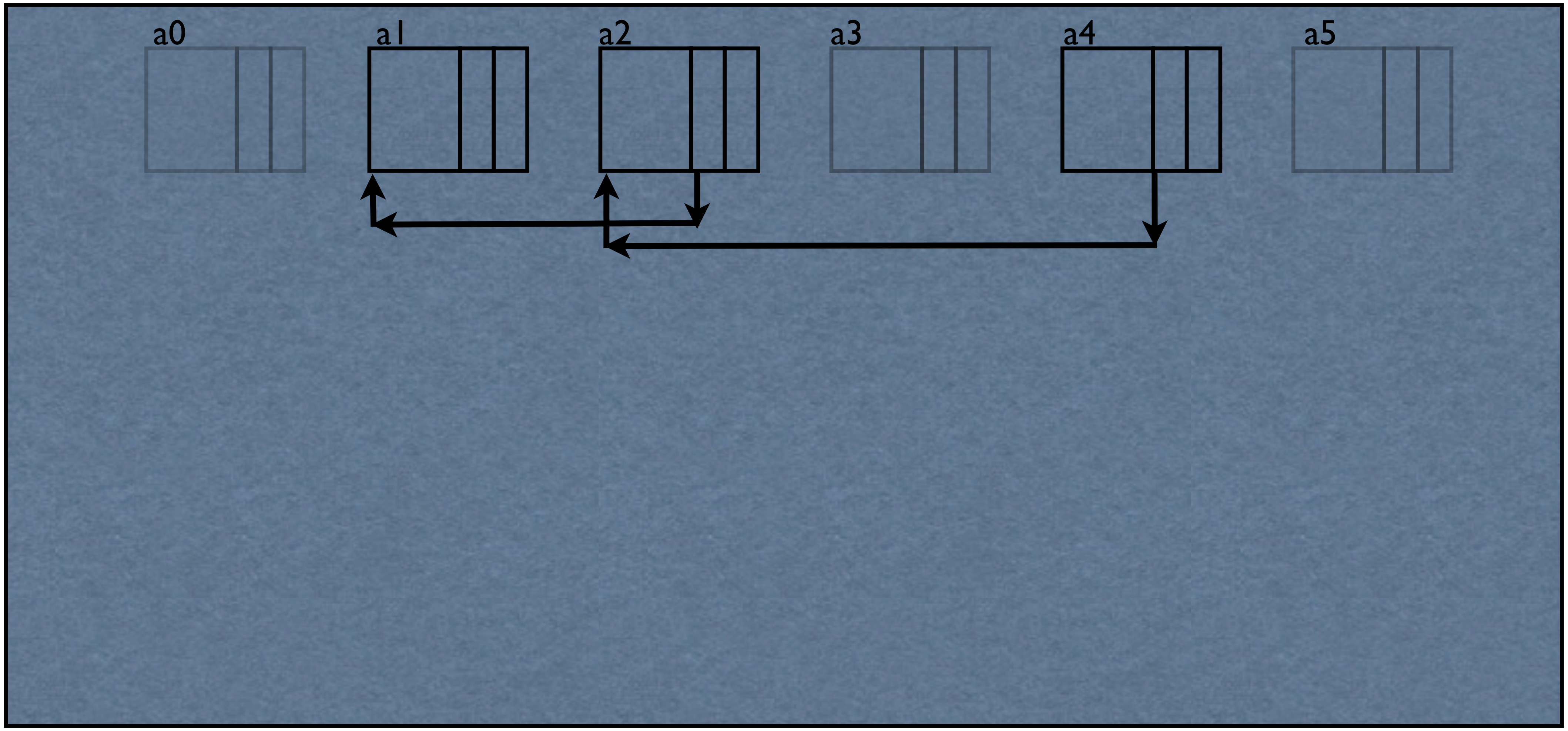
Lisp2

Pass1



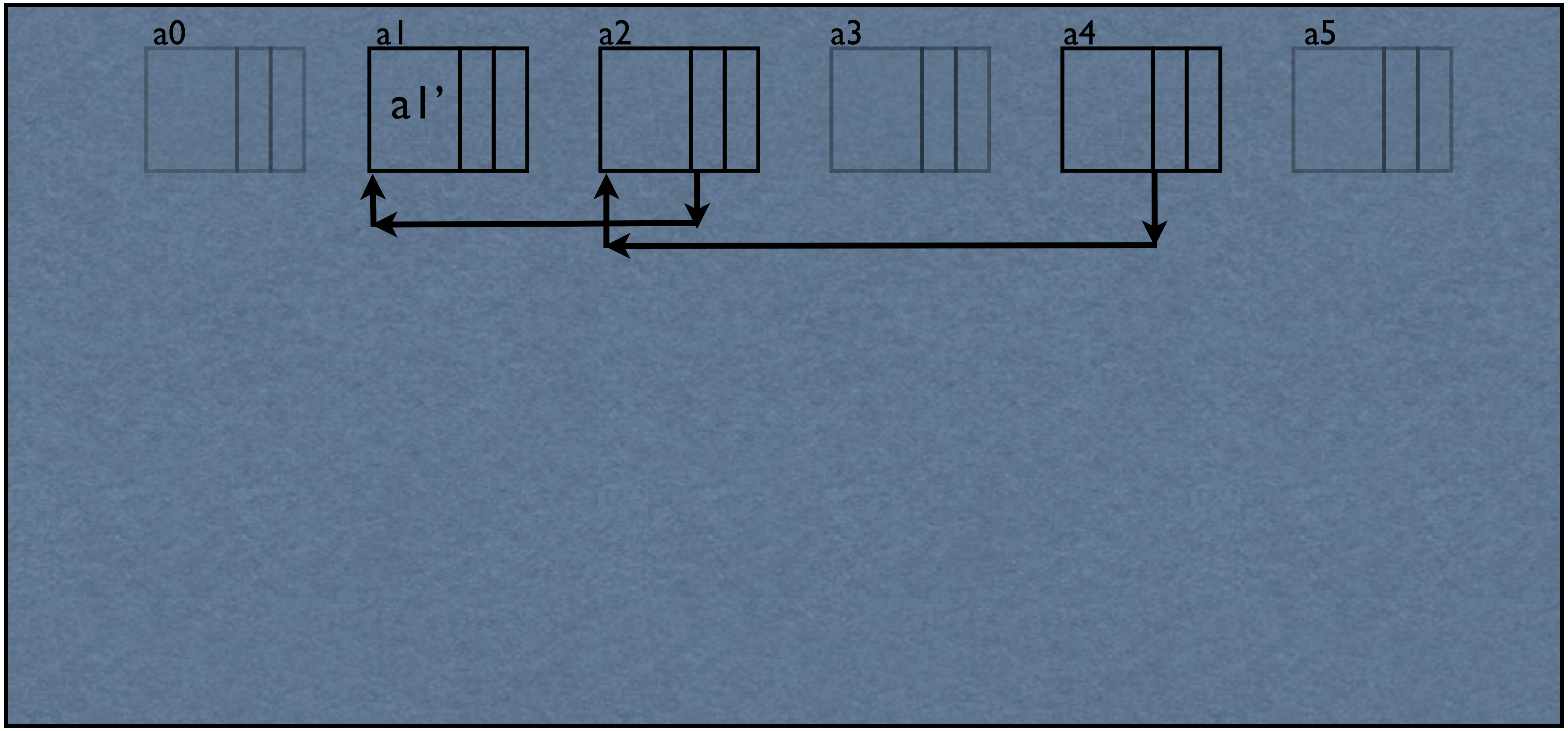
Lisp2

Pass1



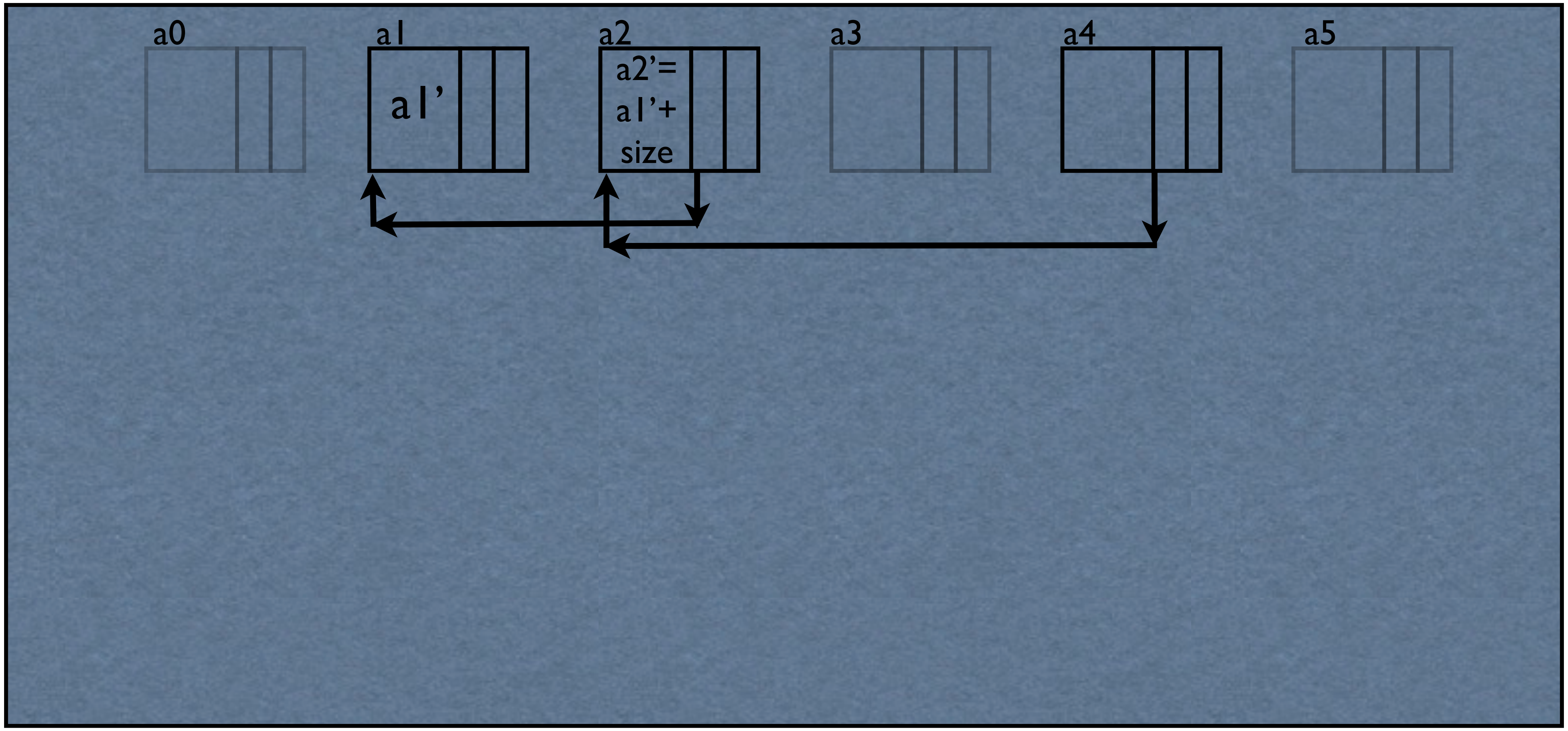
Lisp2

Pass1



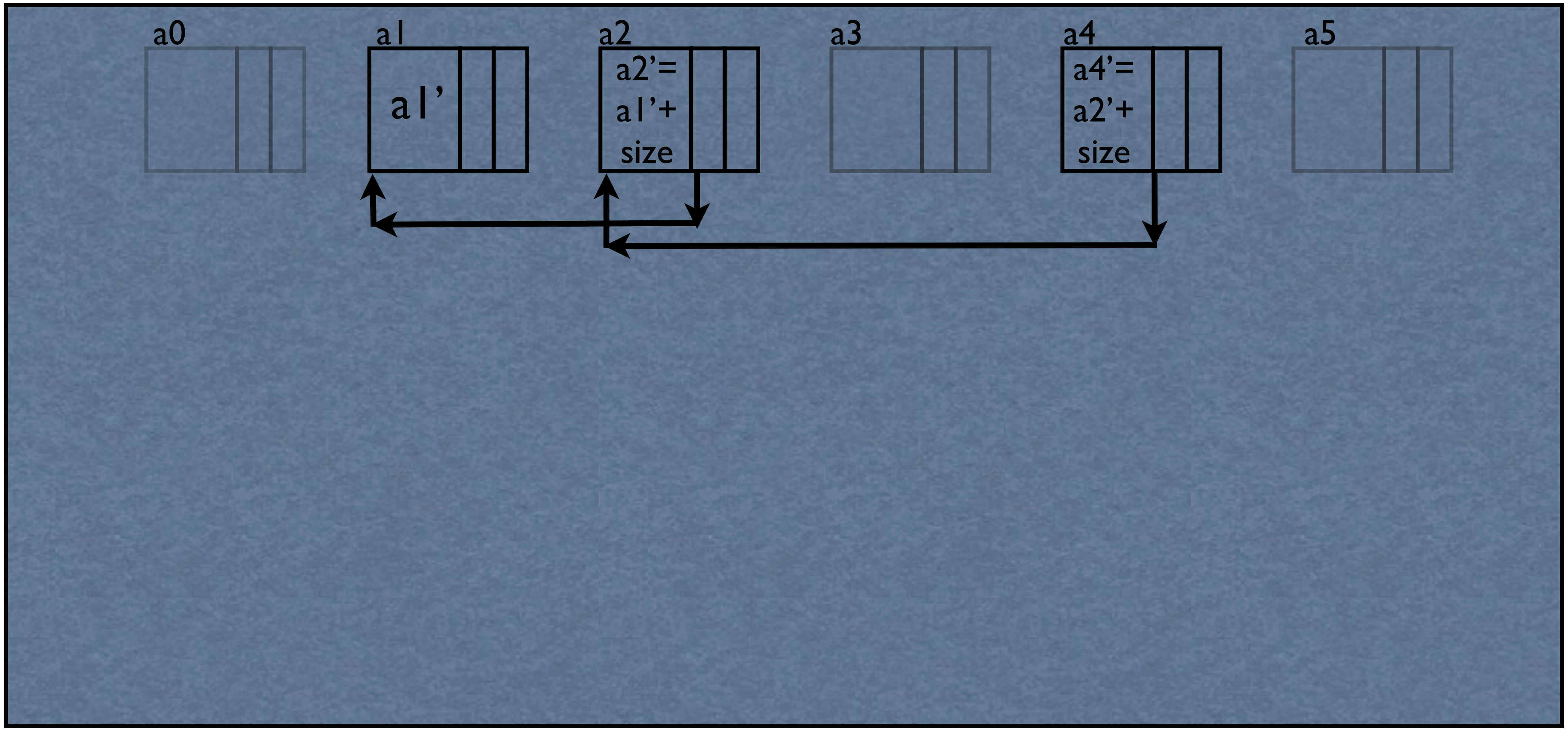
Lisp2

Pass1



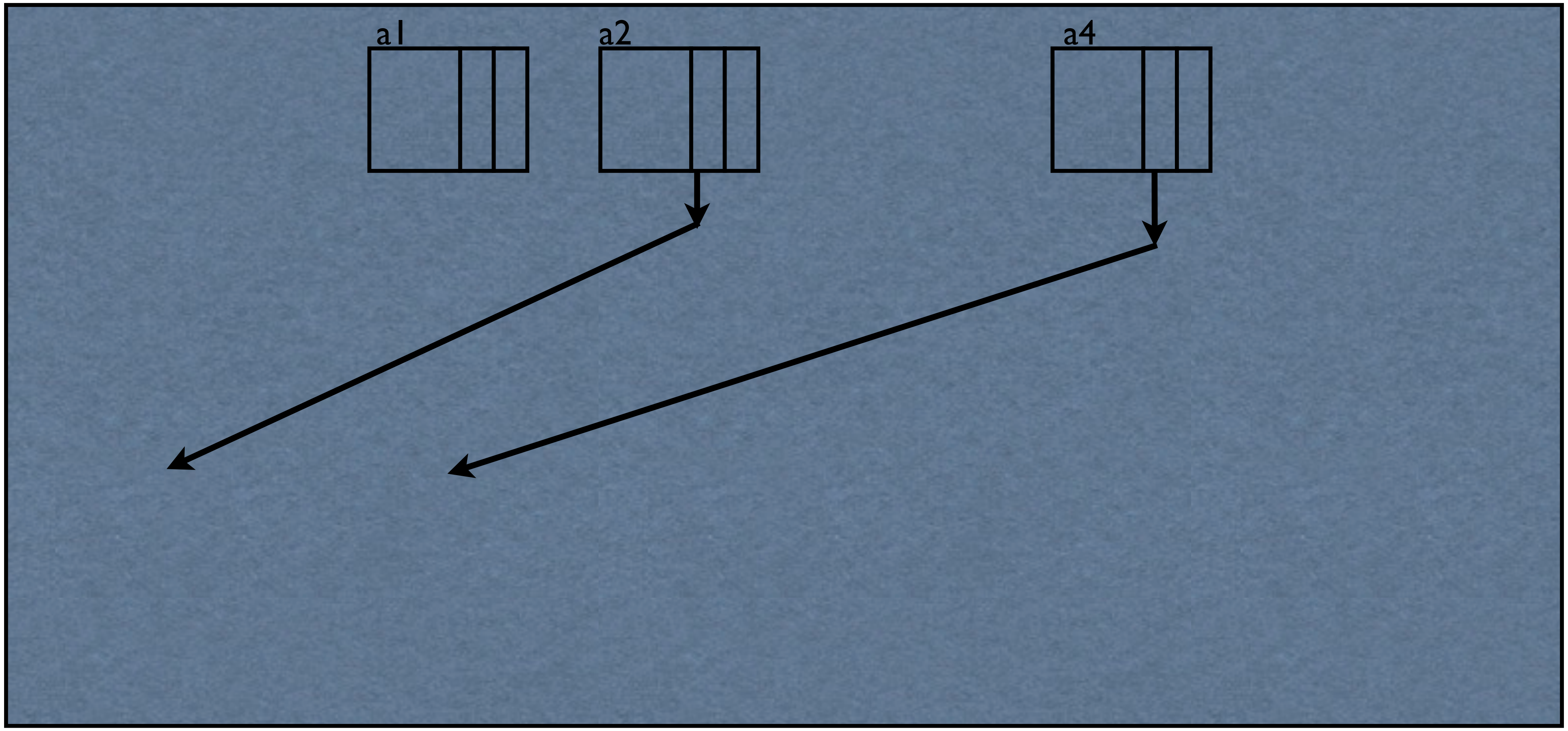
Lisp2

Pass1



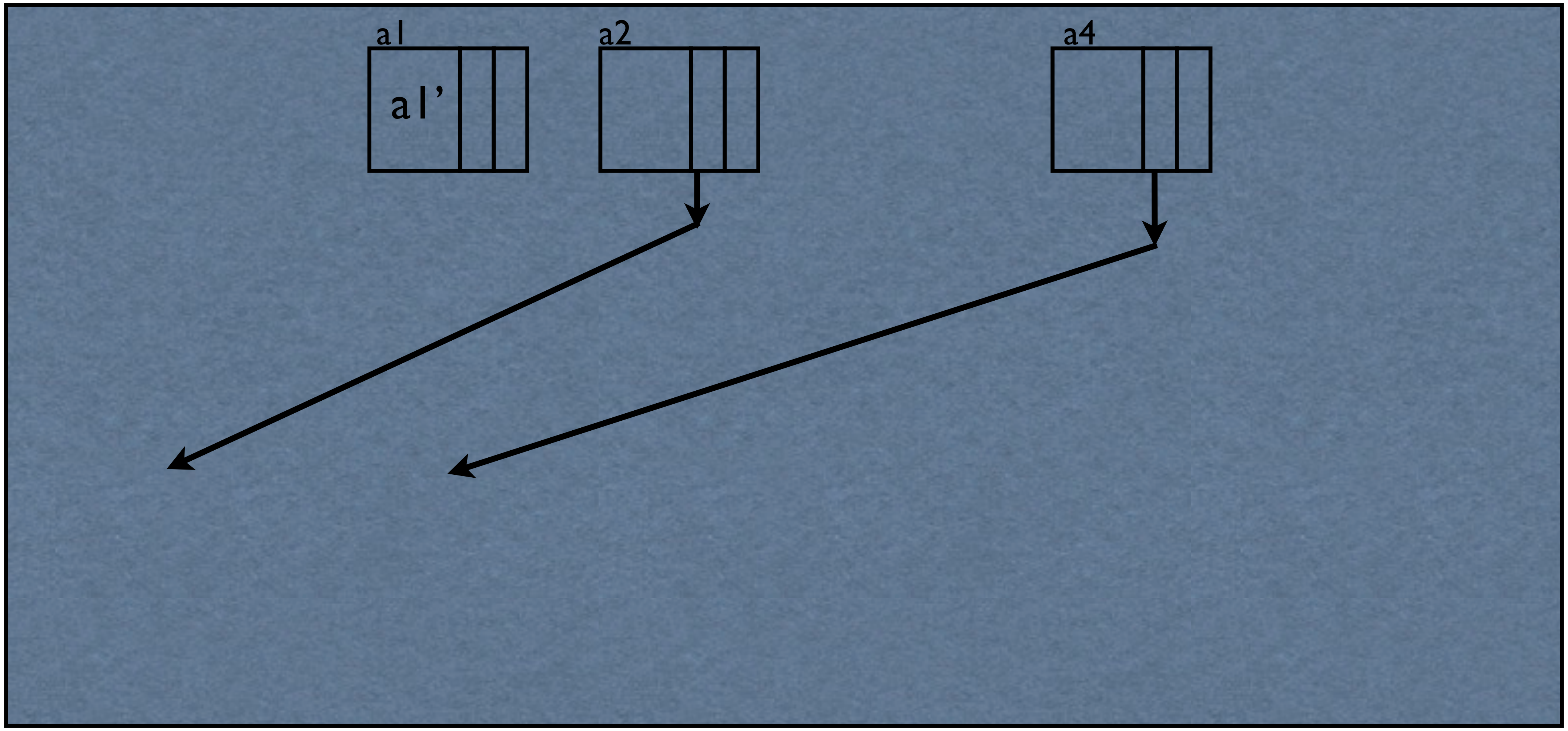
Lisp2

Pass2



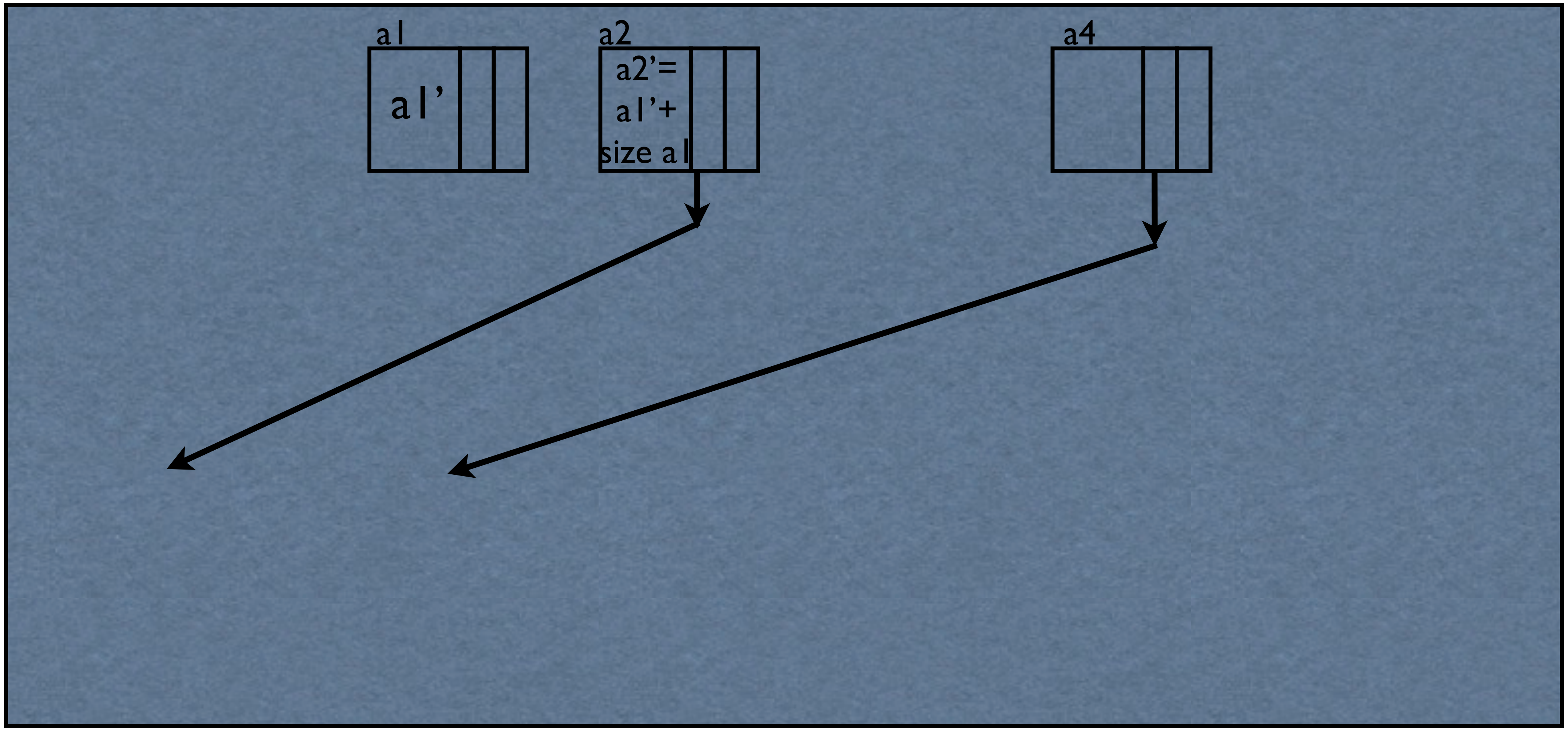
Lisp2

Pass2



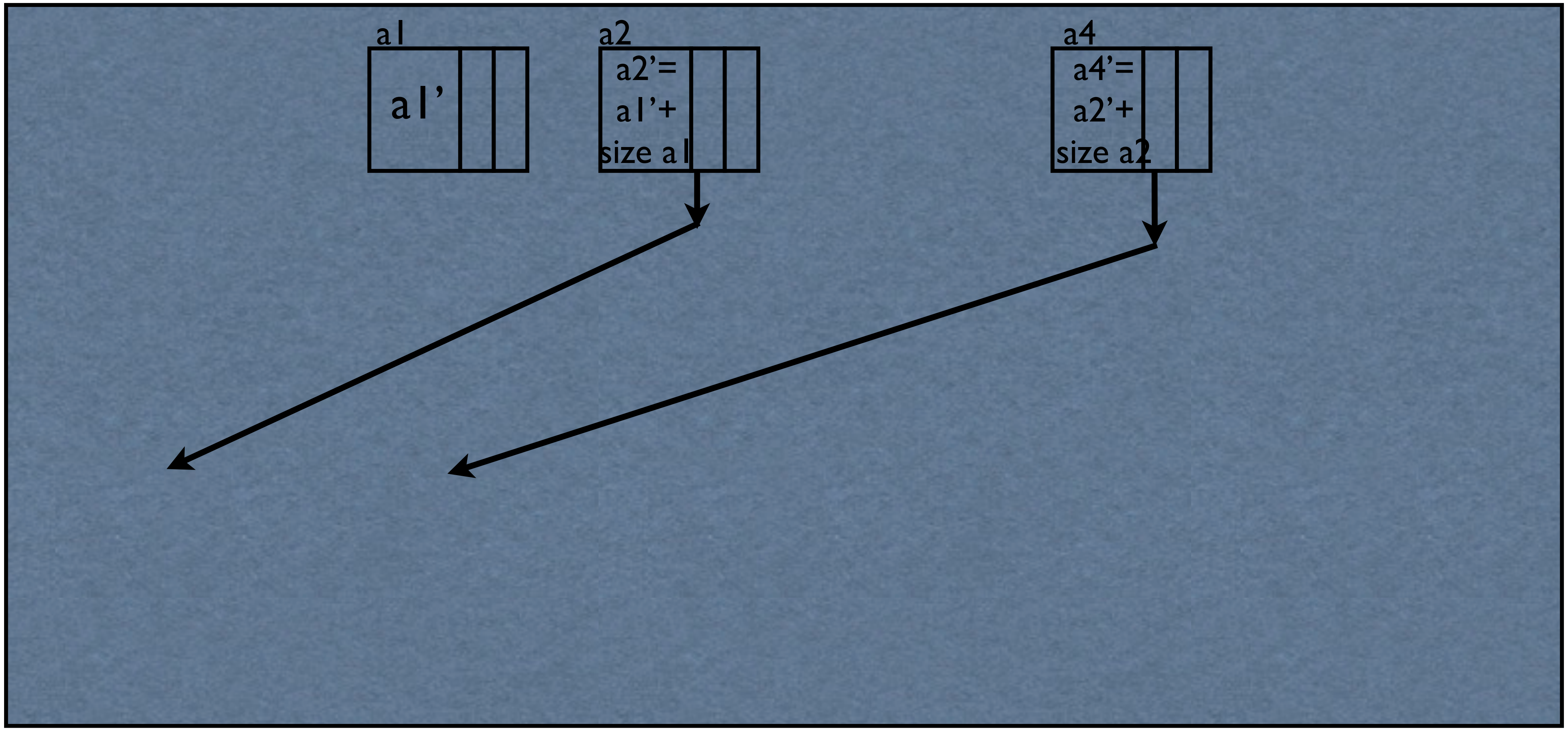
Lisp2

Pass2



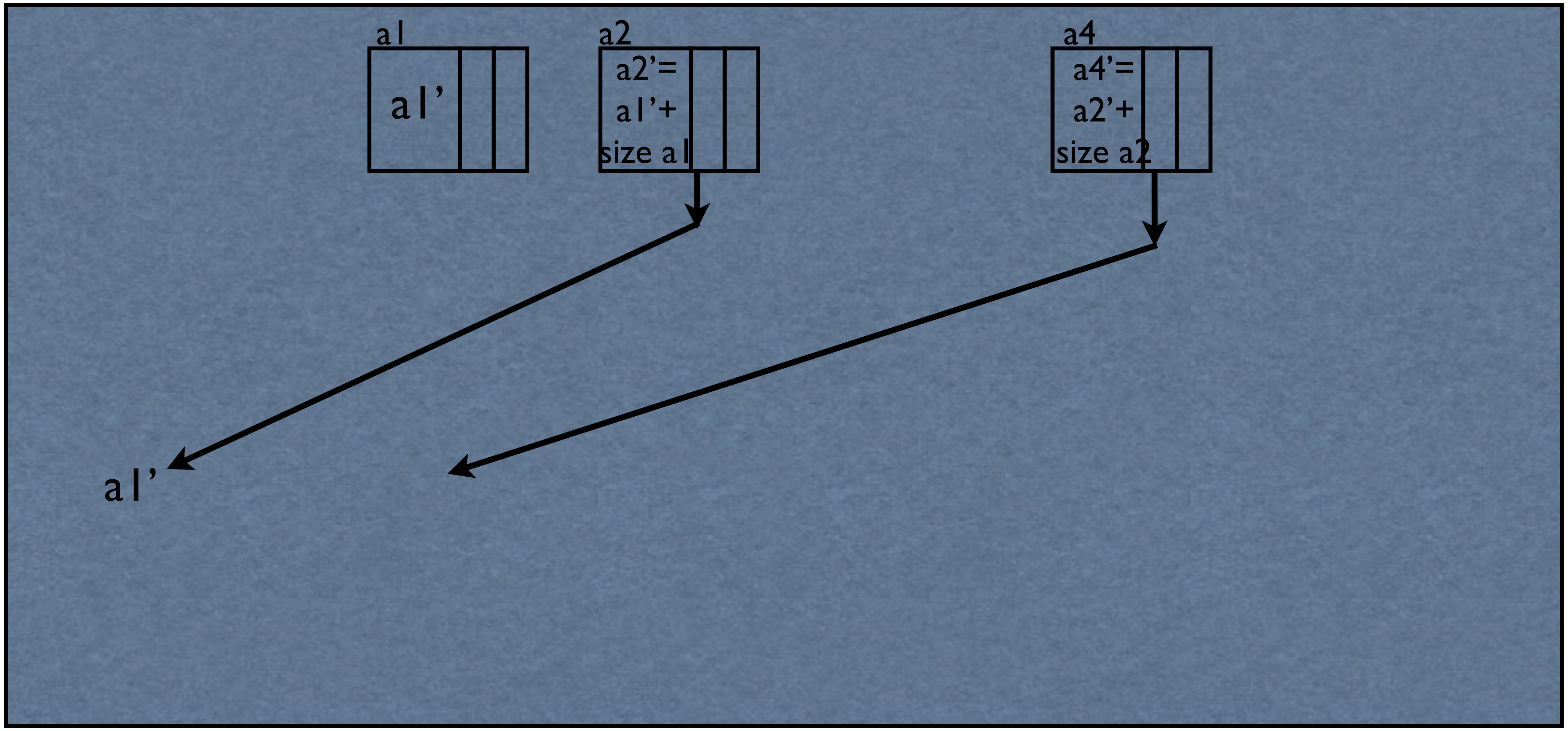
Lisp2

Pass2



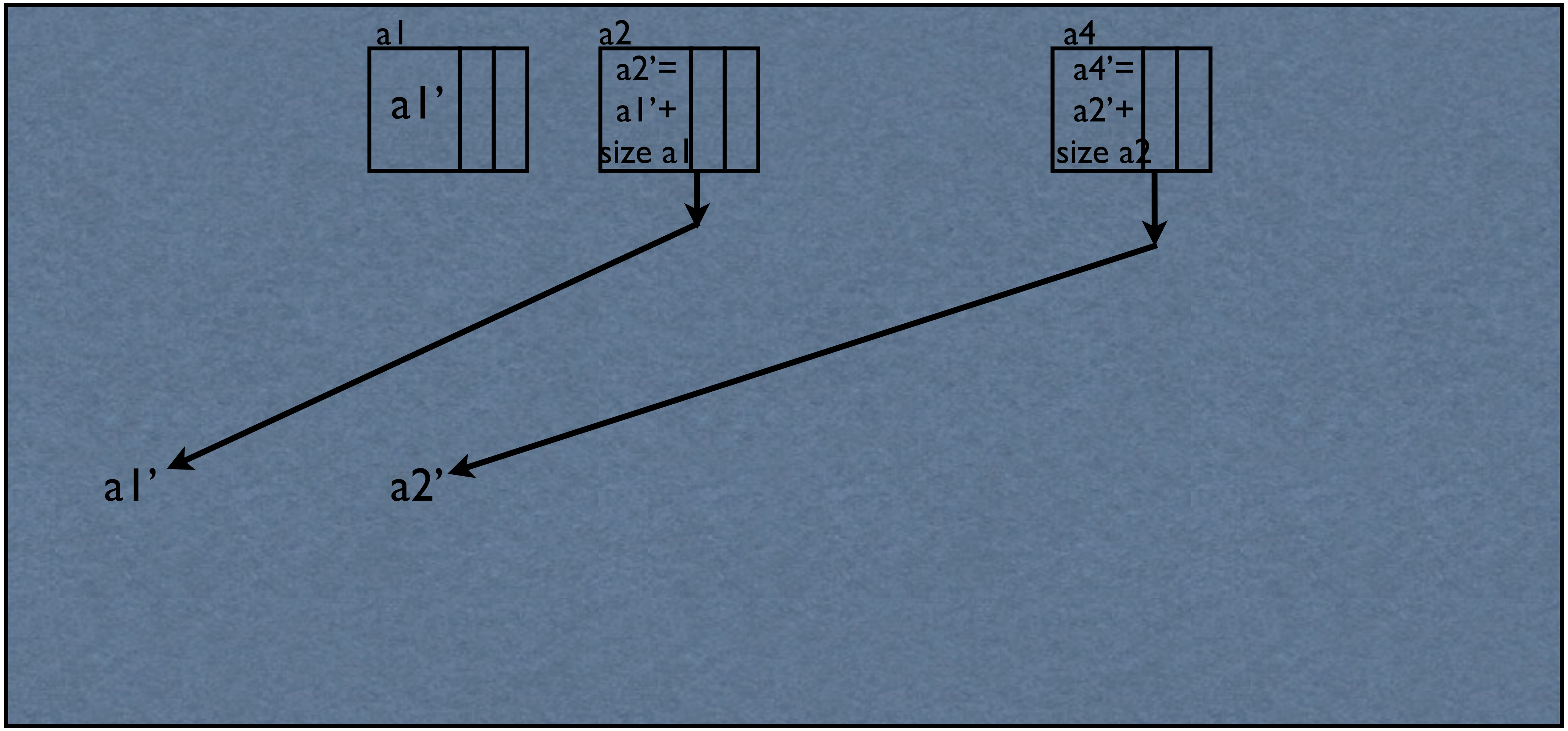
Lisp2

Pass2



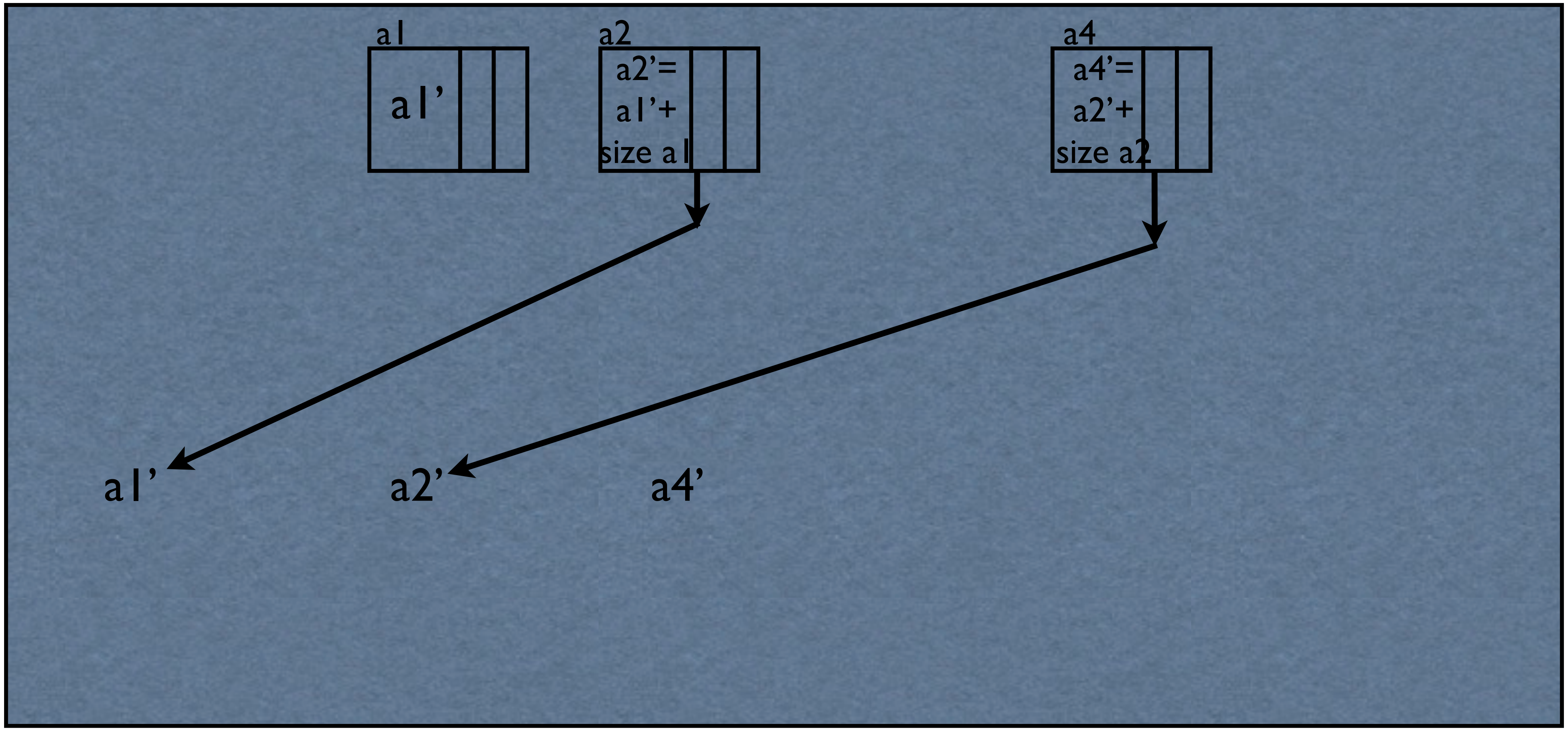
Lisp2

Pass2



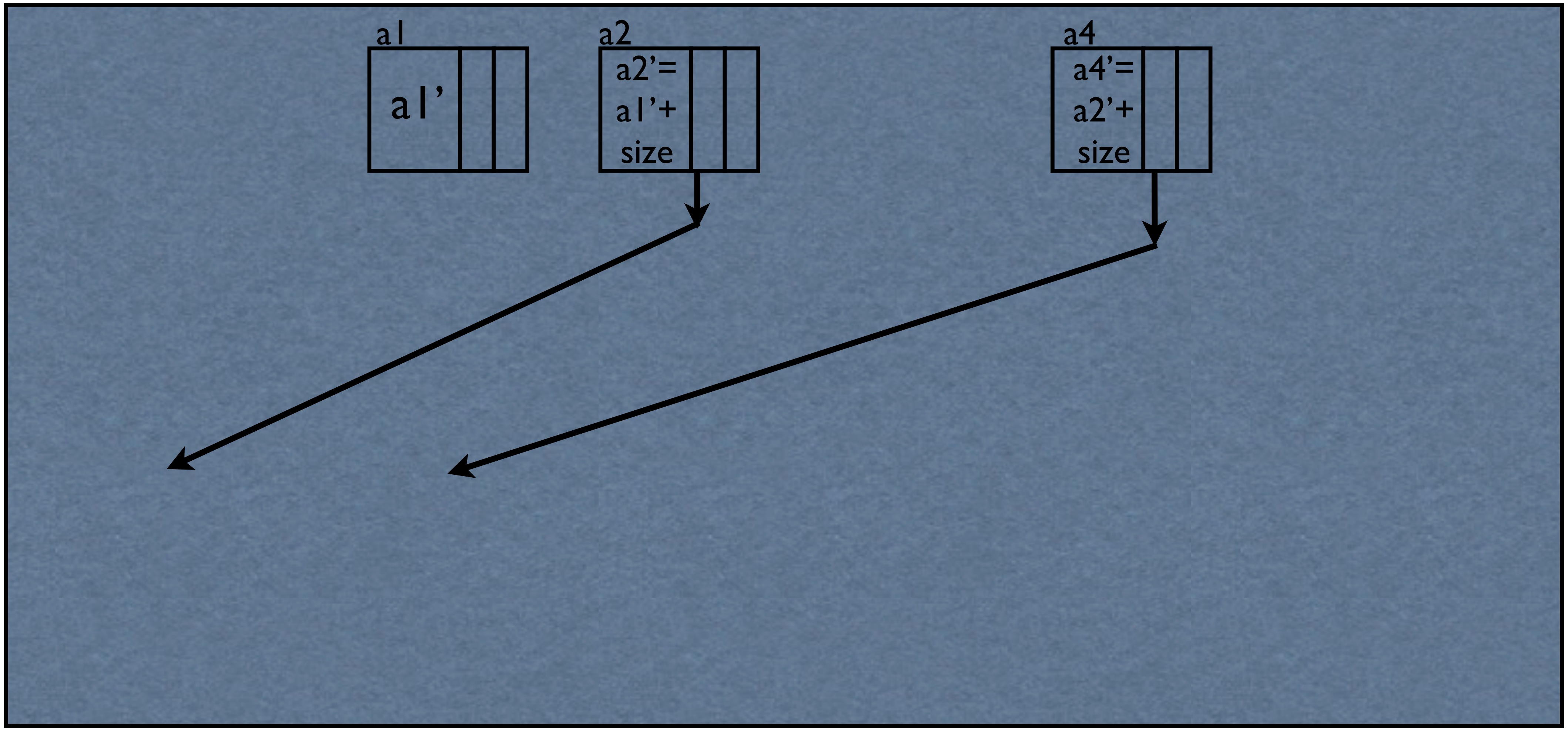
Lisp2

Pass2



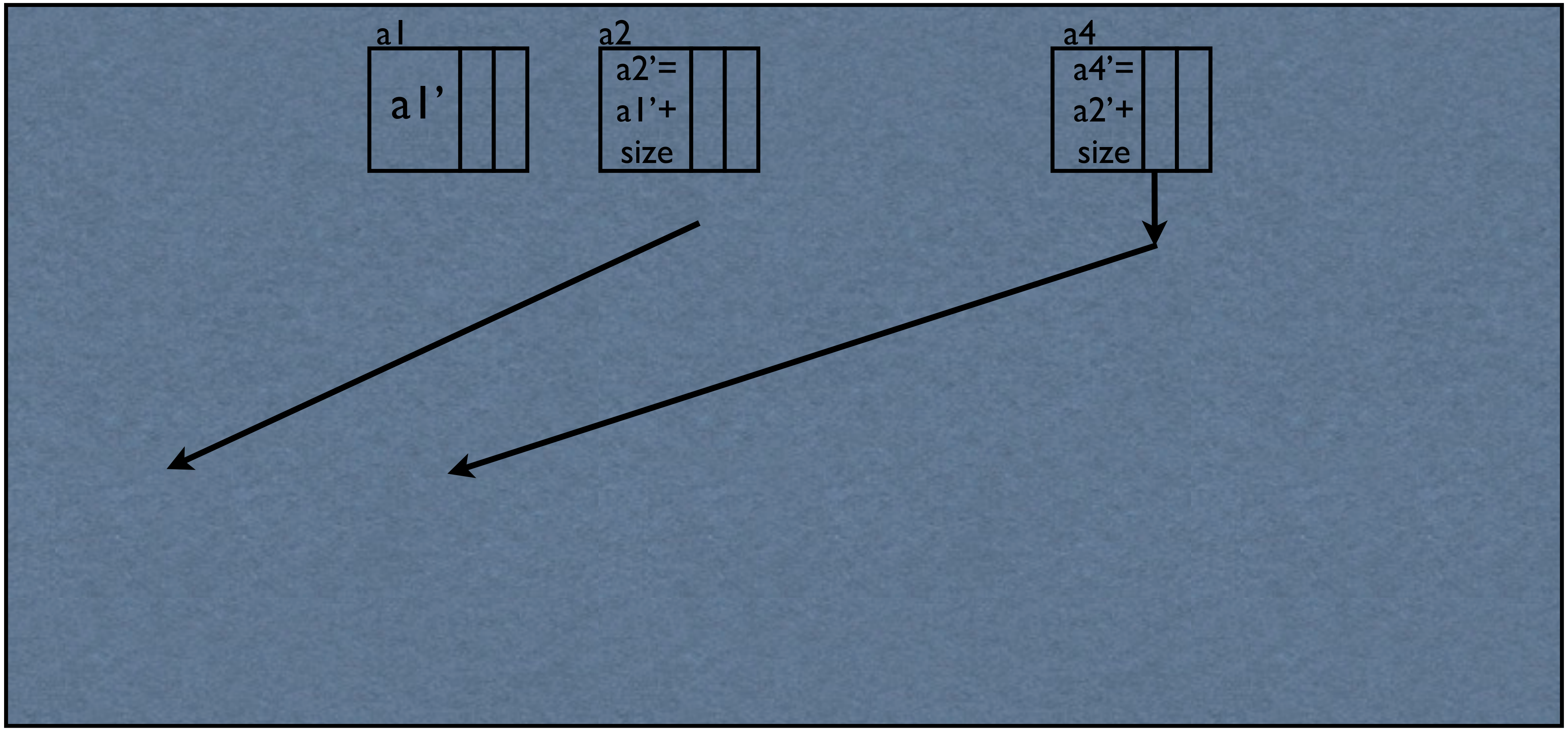
Lisp2

Pass3



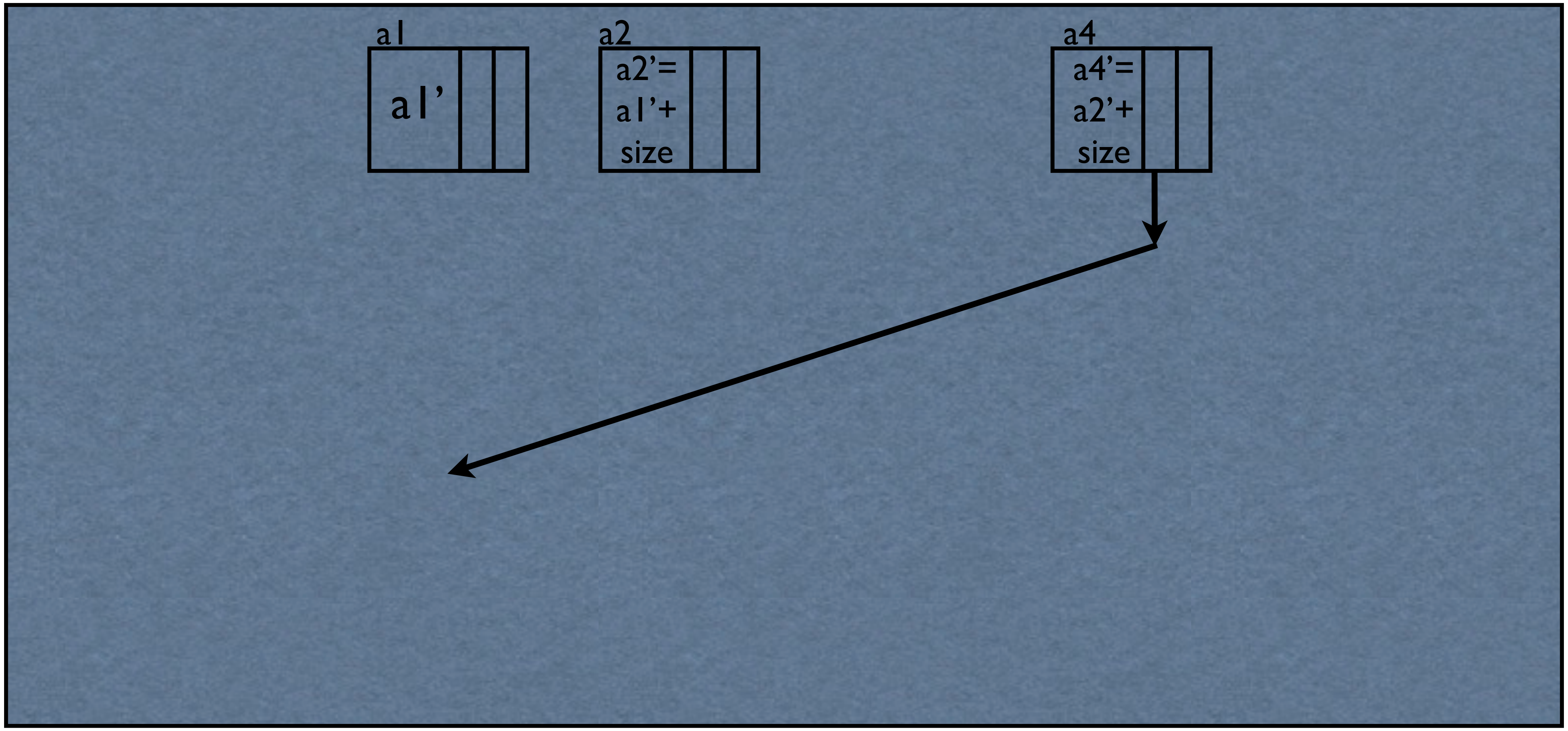
Lisp2

Pass3



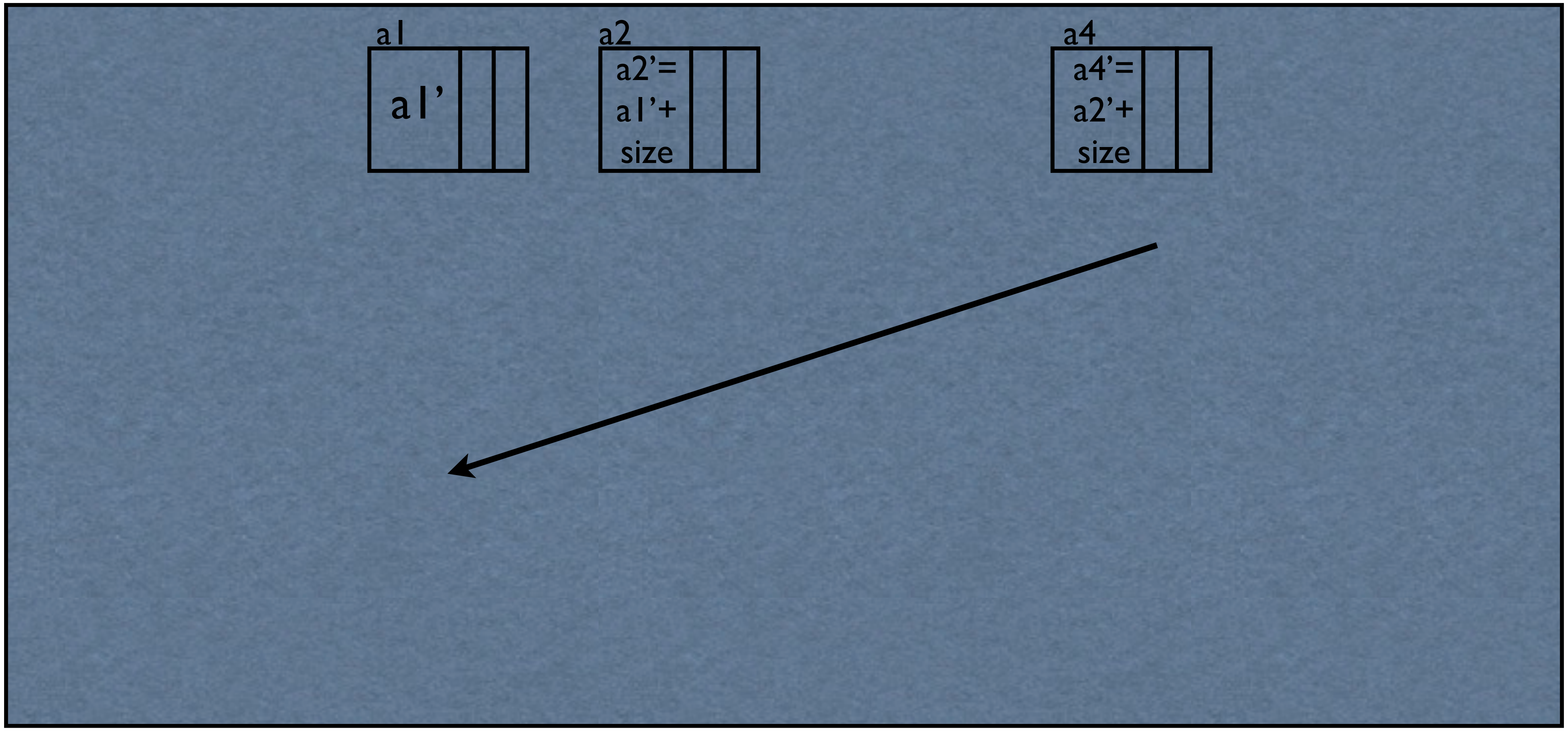
Lisp2

Pass3



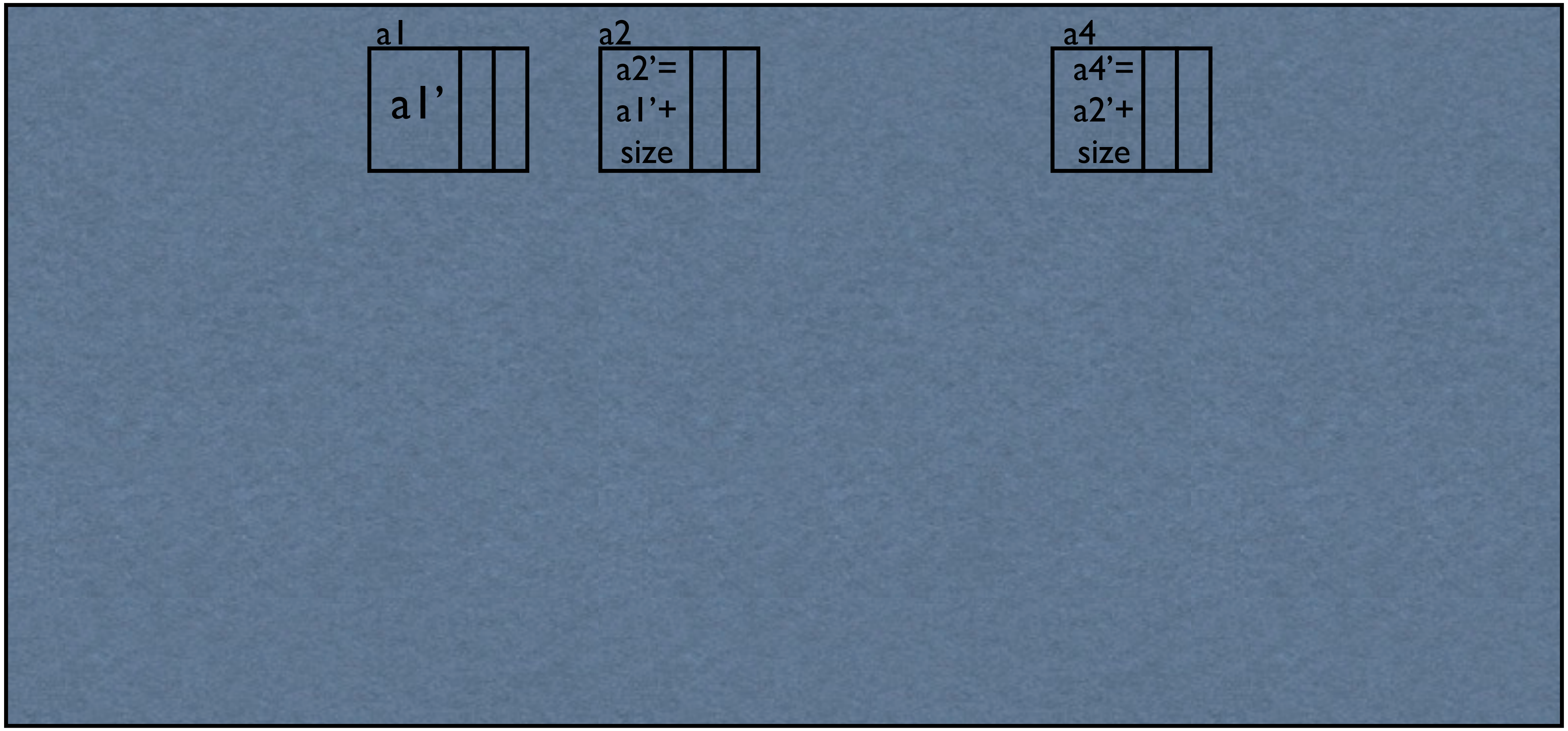
Lisp2

Pass3



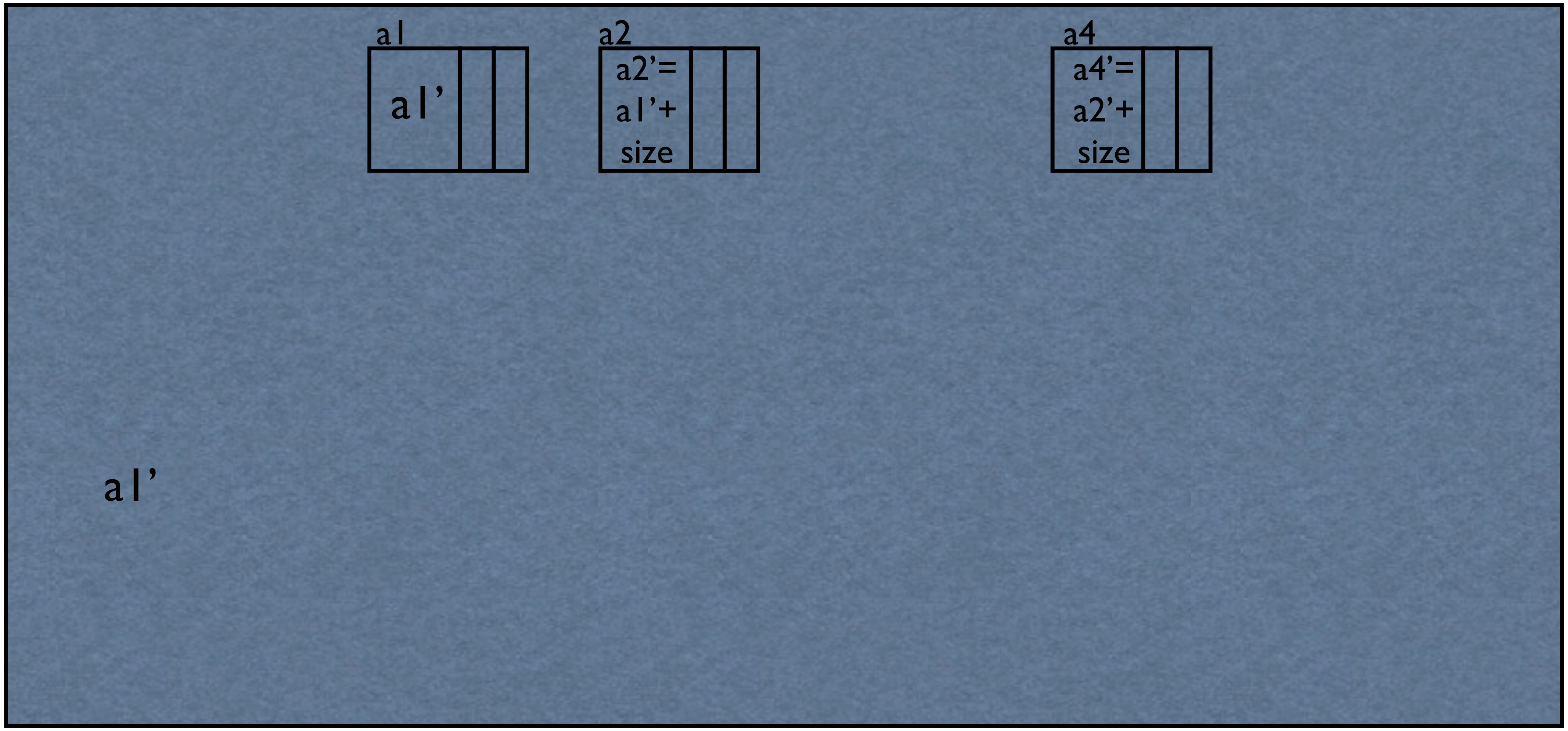
Lisp2

Pass3



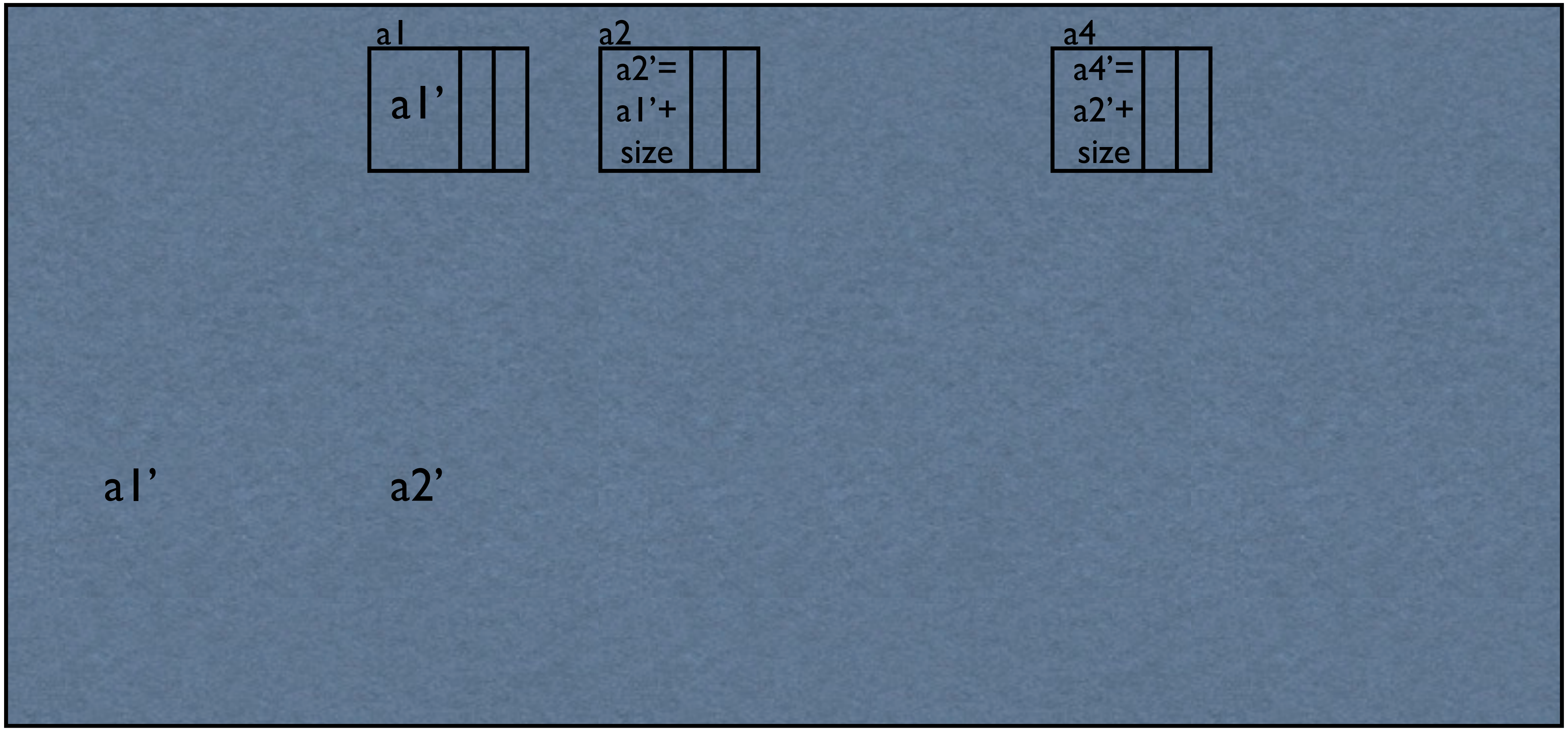
Lisp2

Pass3



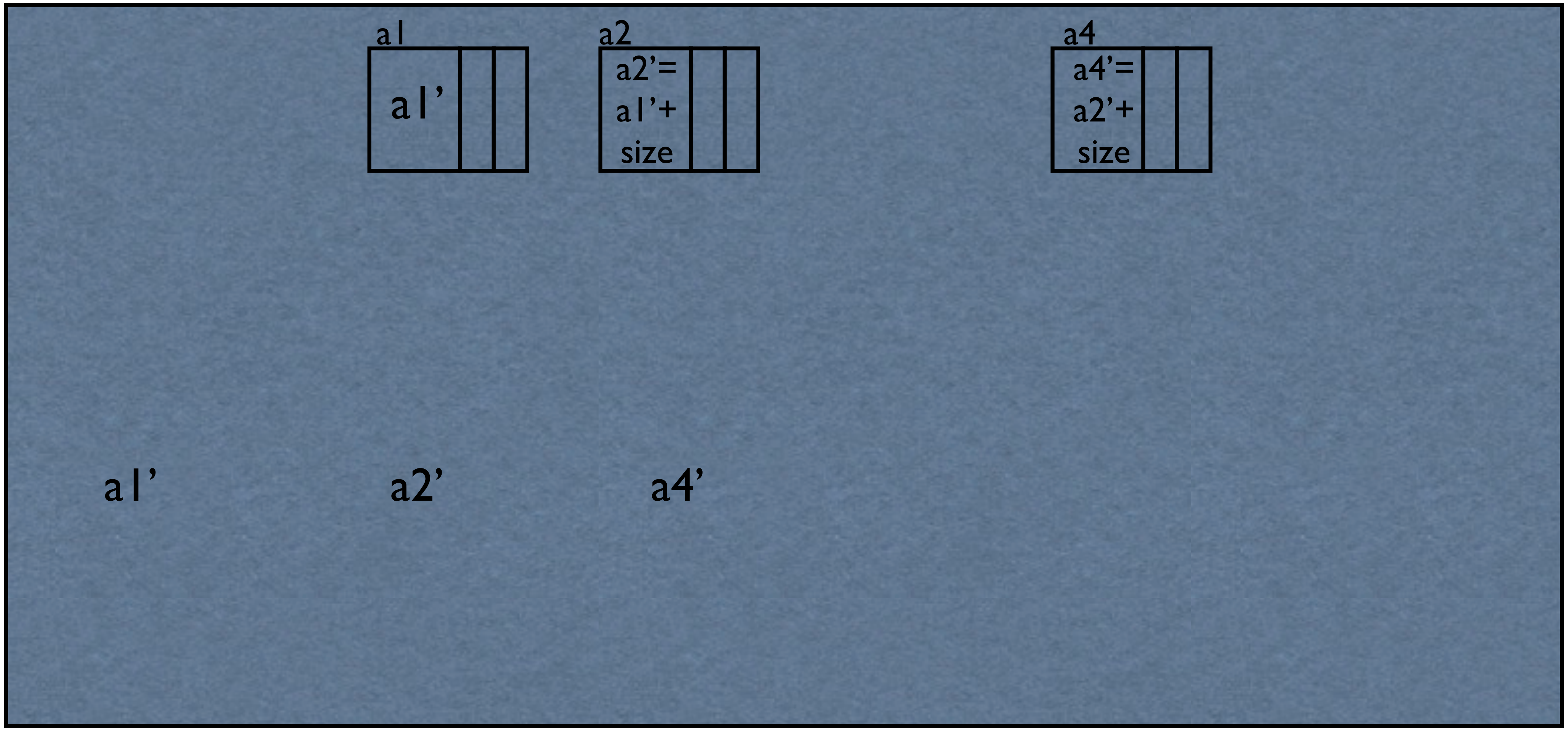
Lisp2

Pass3



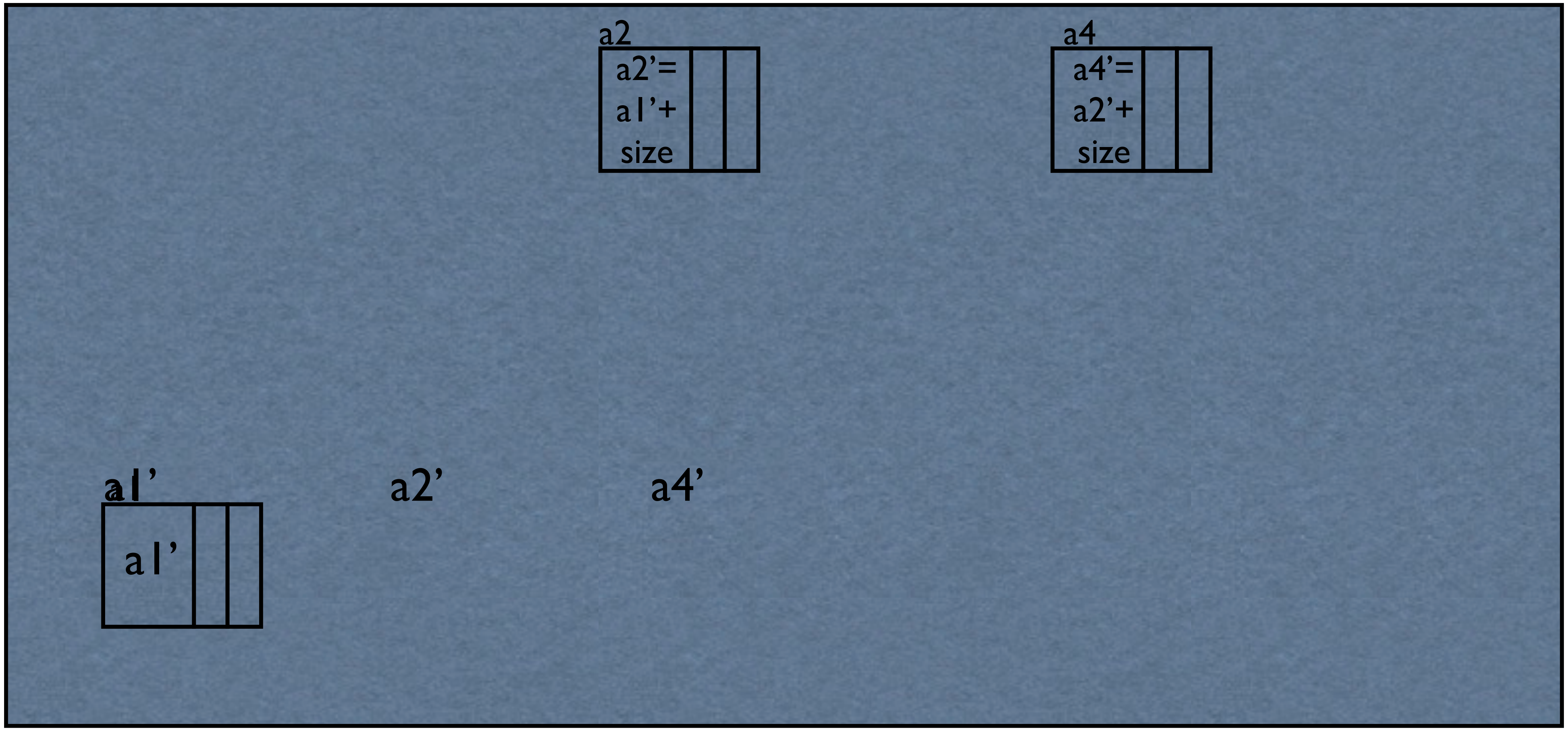
Lisp2

Pass3



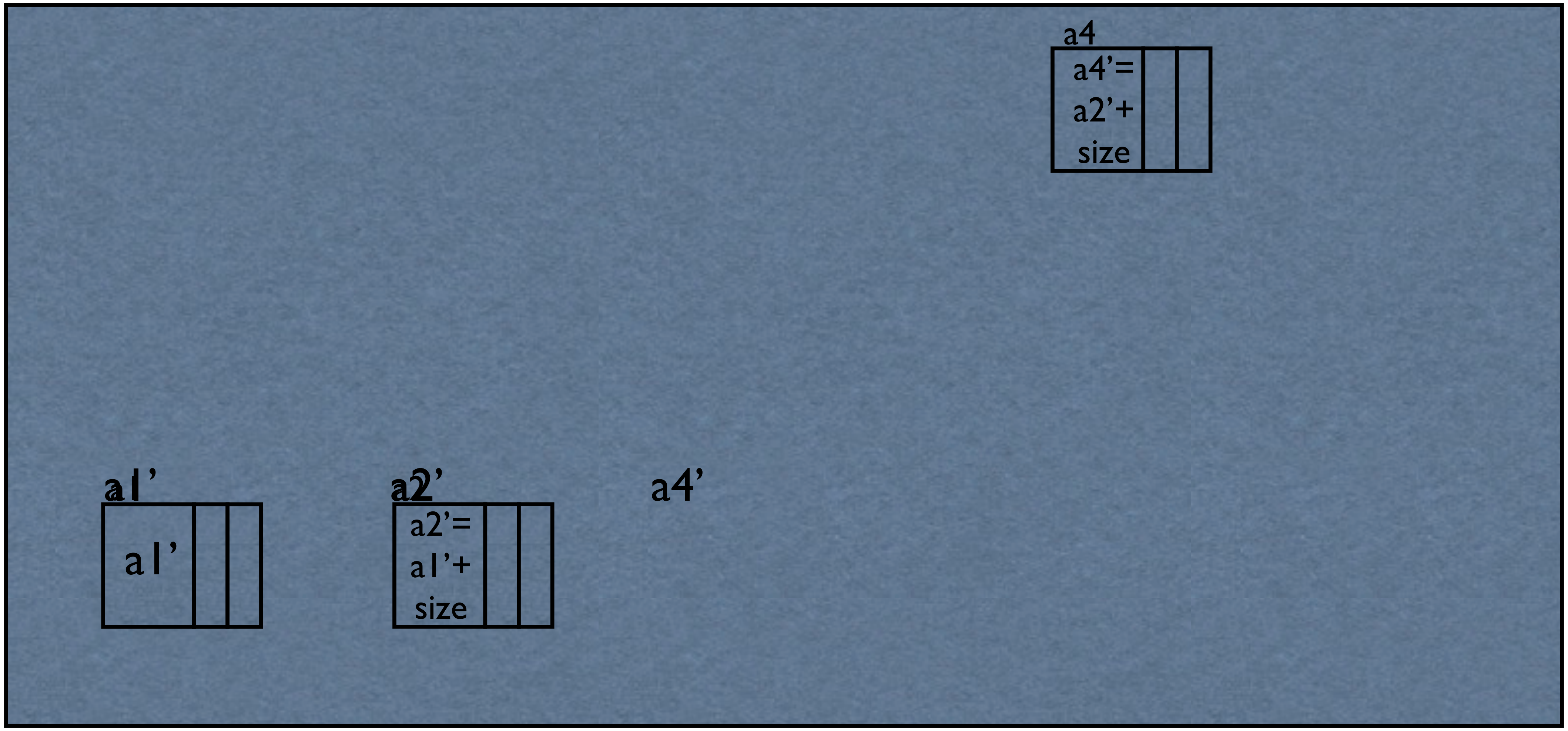
Lisp2

Pass3



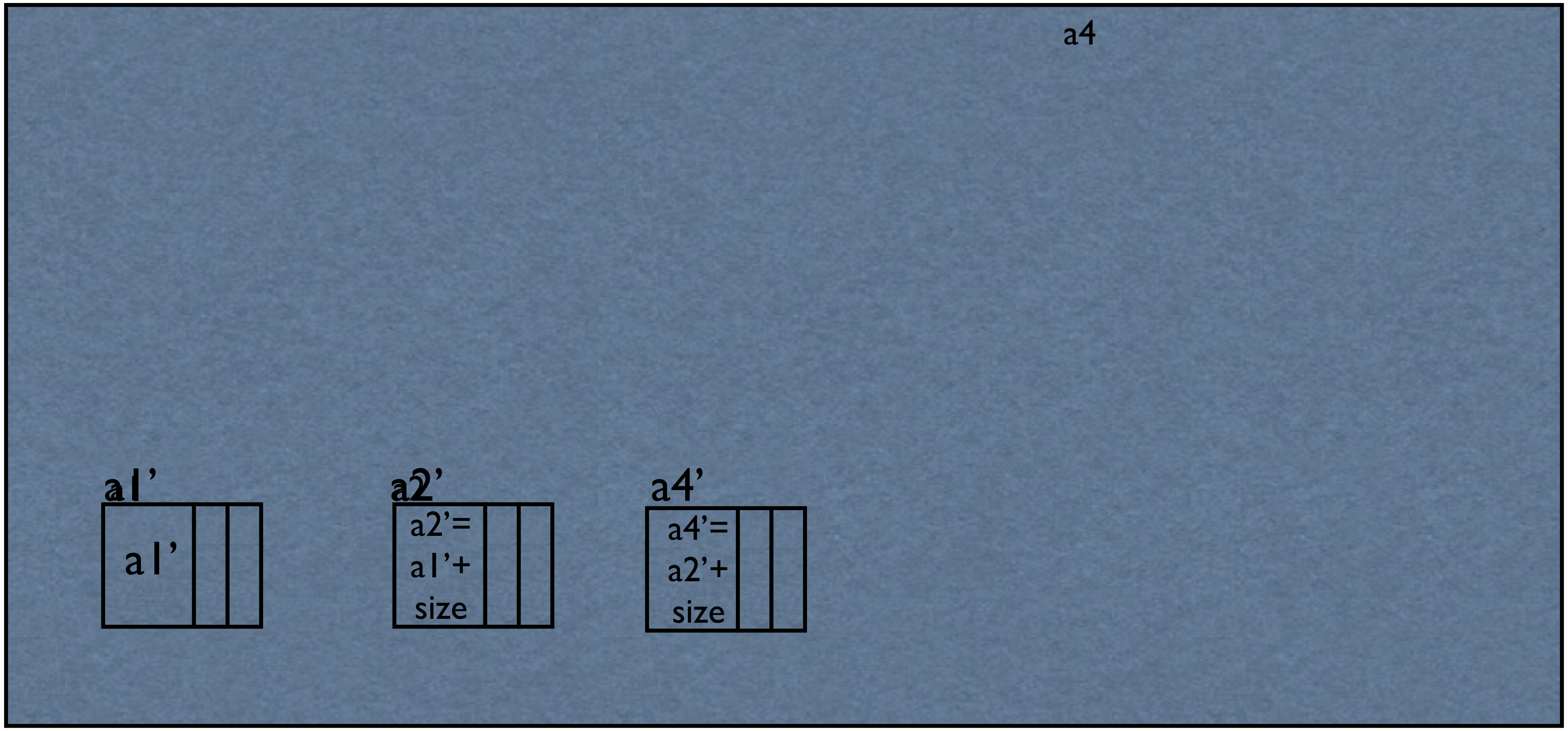
Lisp2

Pass3

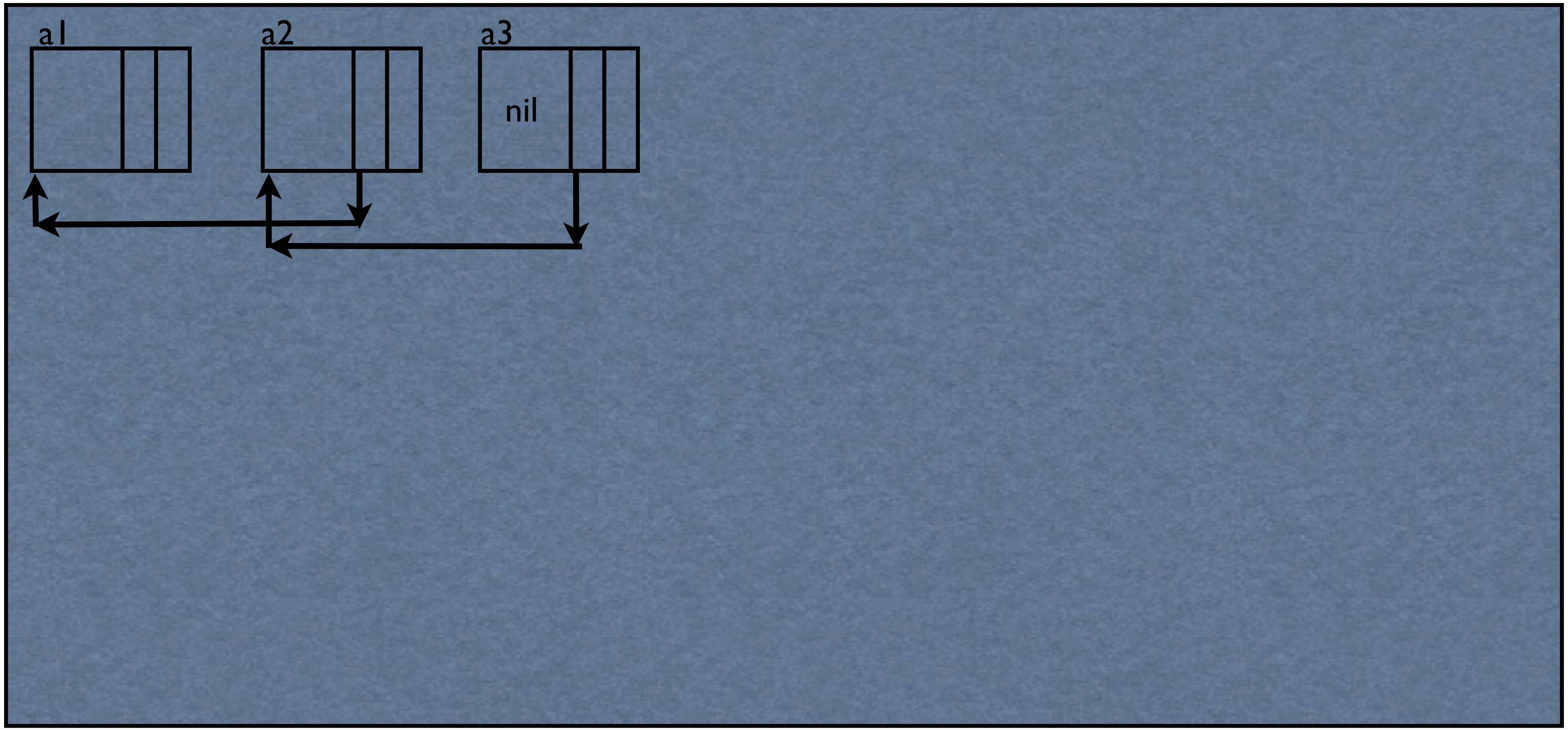


Lisp2

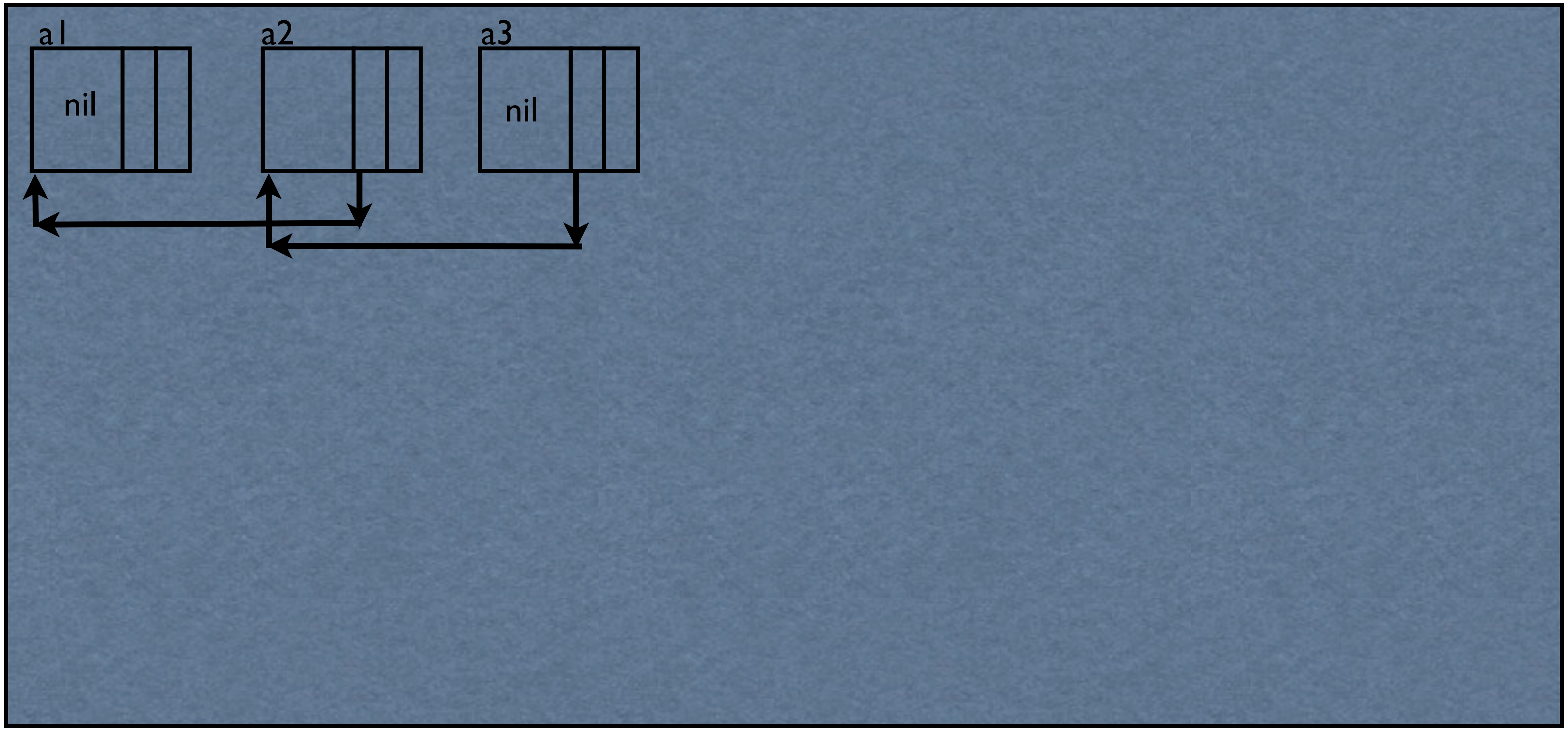
Pass3



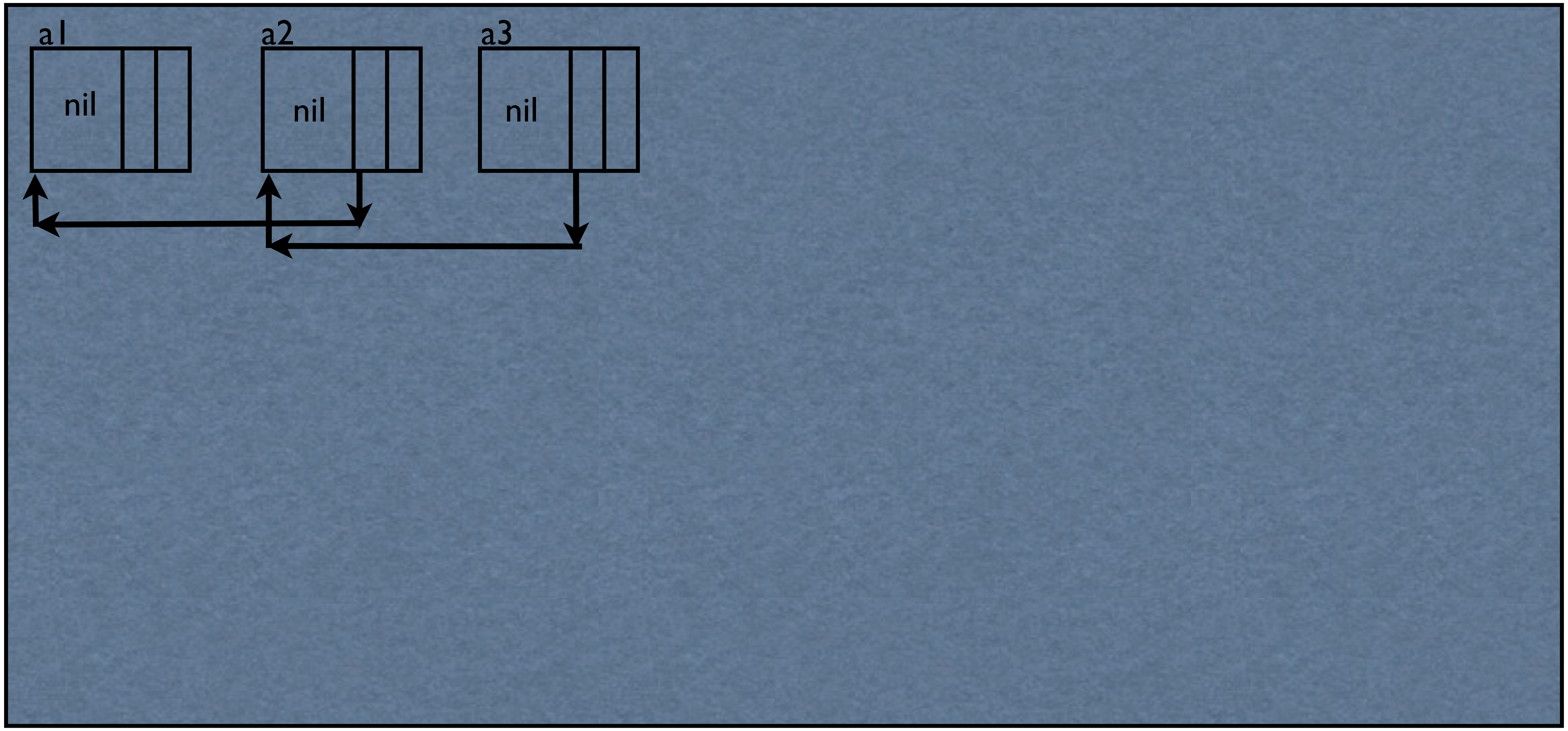
Lisp2..Final



Lisp2..Final



Lisp2..Final



Lisp2 .. Summary

- Requires 1 extra word in each object for temp pointer. (even when the object is not live)
- Compaction is done in 3 phases:

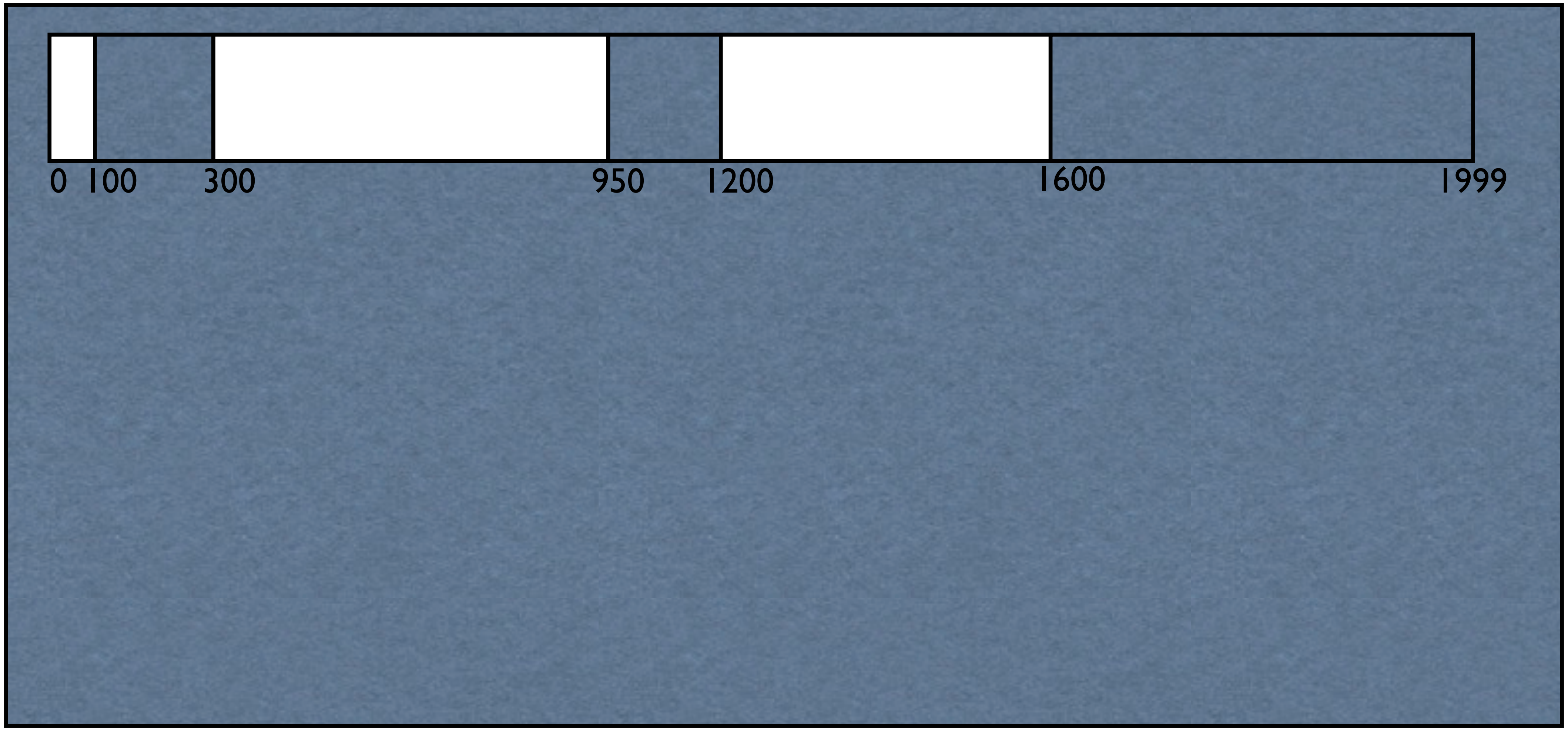
1. Traverse the objects, sorted by address
    - Compute new address of each live object
    - free_ptr=0; free_ptr+=free_ptr+size of live object
2. Update Pointer fields.
3. Sliding Compaction

Table Compactors

- We need to save the overhead due to temp pointers.
- Using inactive cells to store readjustments.

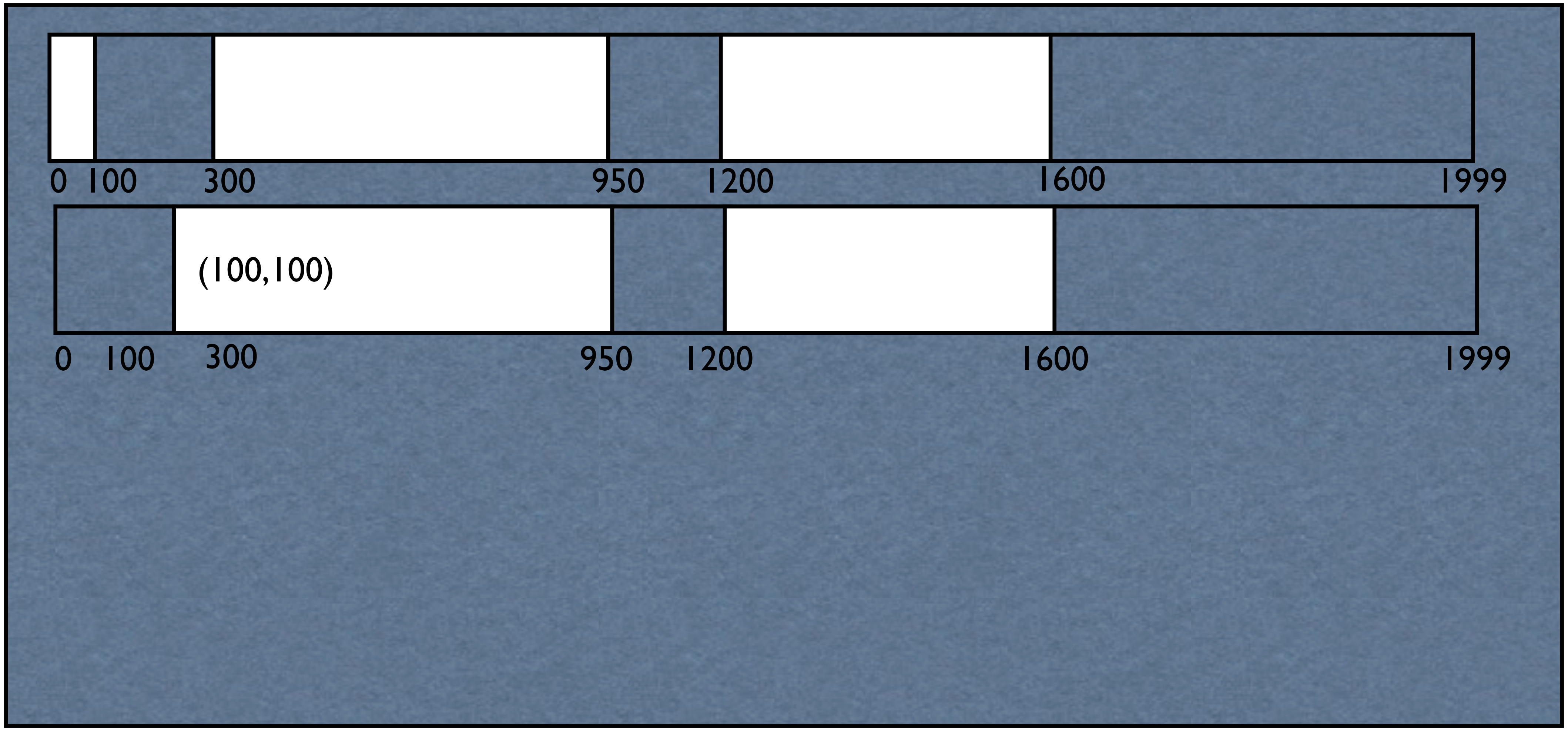
Break Table

Phase I



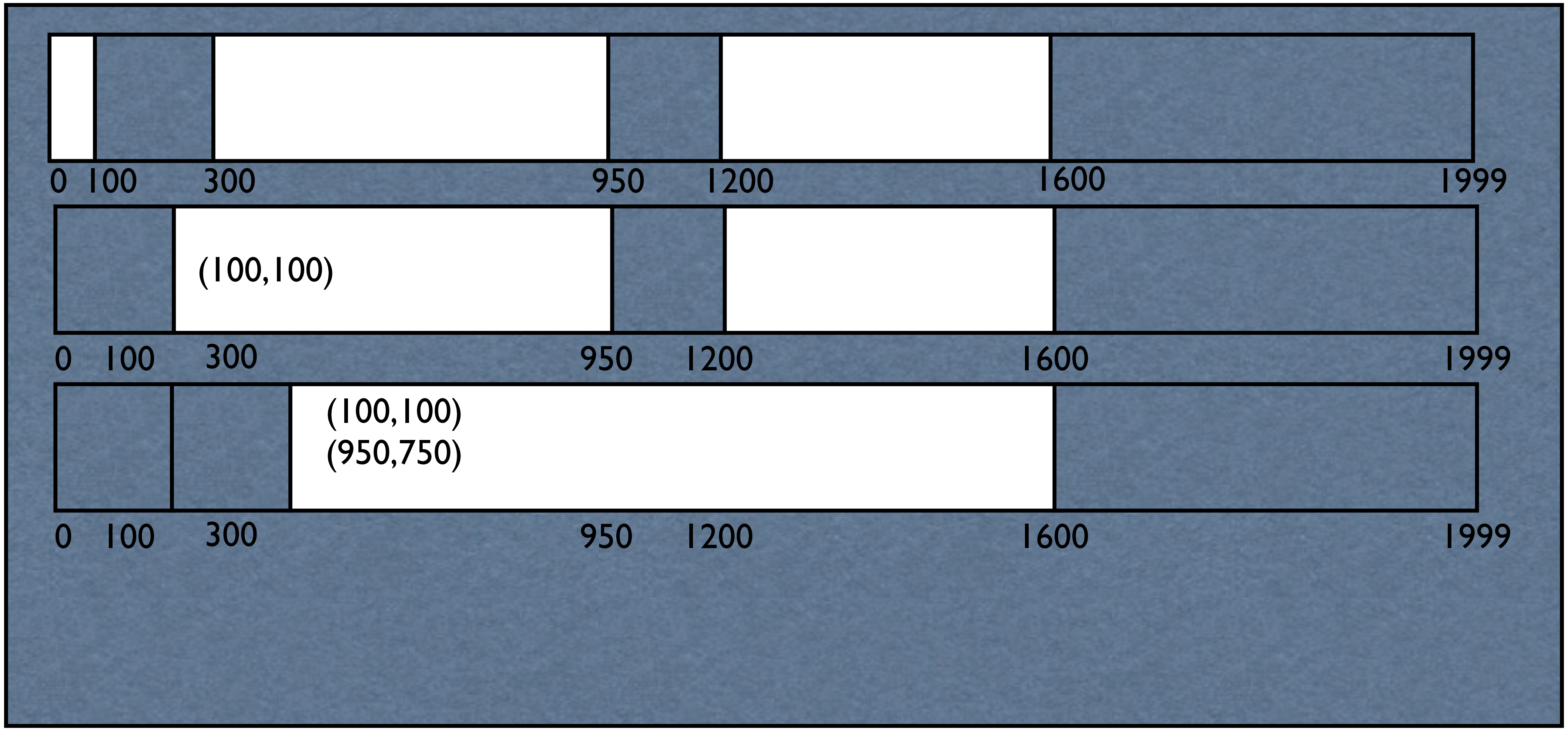
Break Table

Phase I



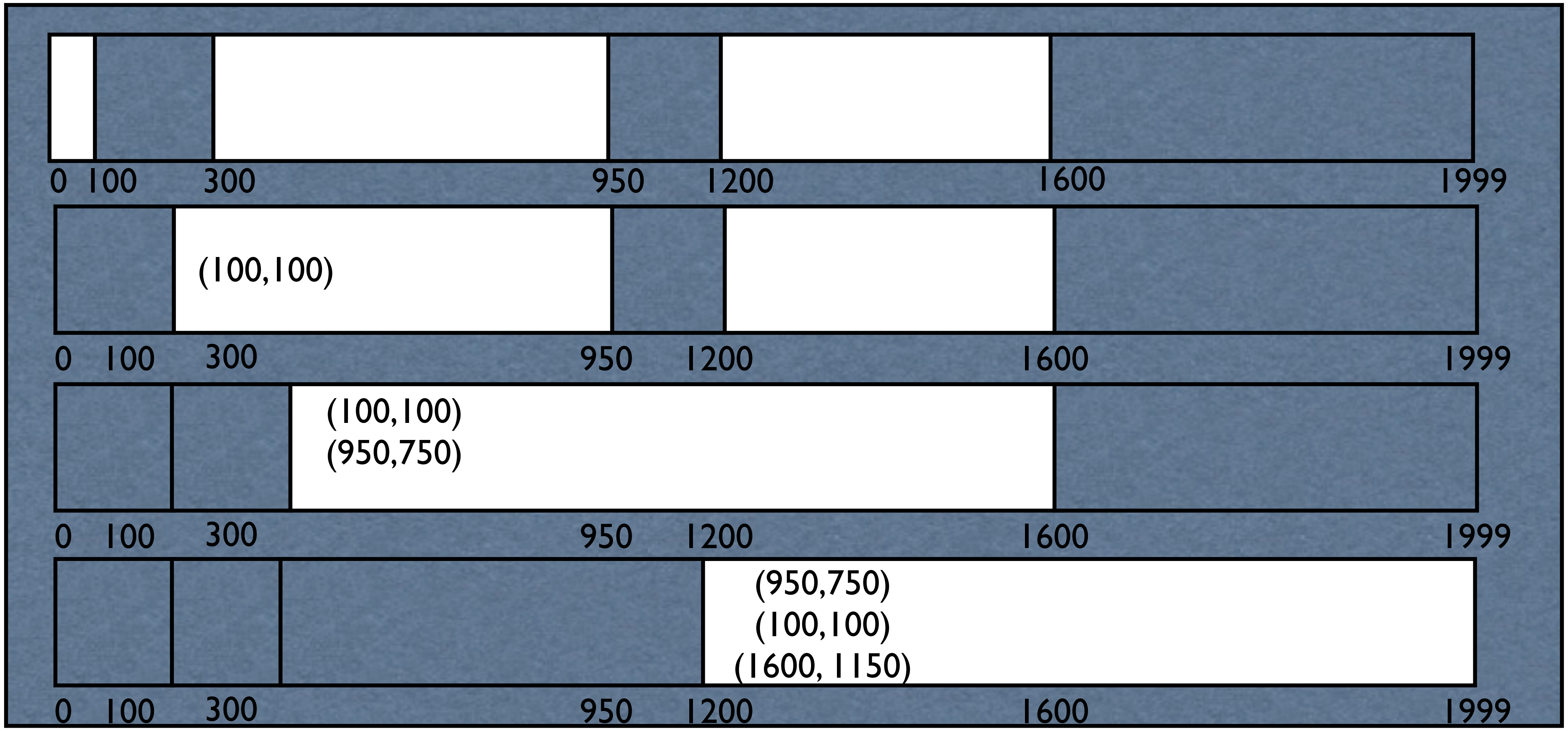
Break Table

Phase I



Break Table

Phase I



Break Table

- Rolling back causes it to become unsorted.
- Need another phase just to sort the BT.

Break Table

- Phase3 to fix the pointers.

  1. Search through the BT table and determine the adjacent pairs(a, s) and (a', s') such that a <= p < a'

  2. readjusted value should be p - s.

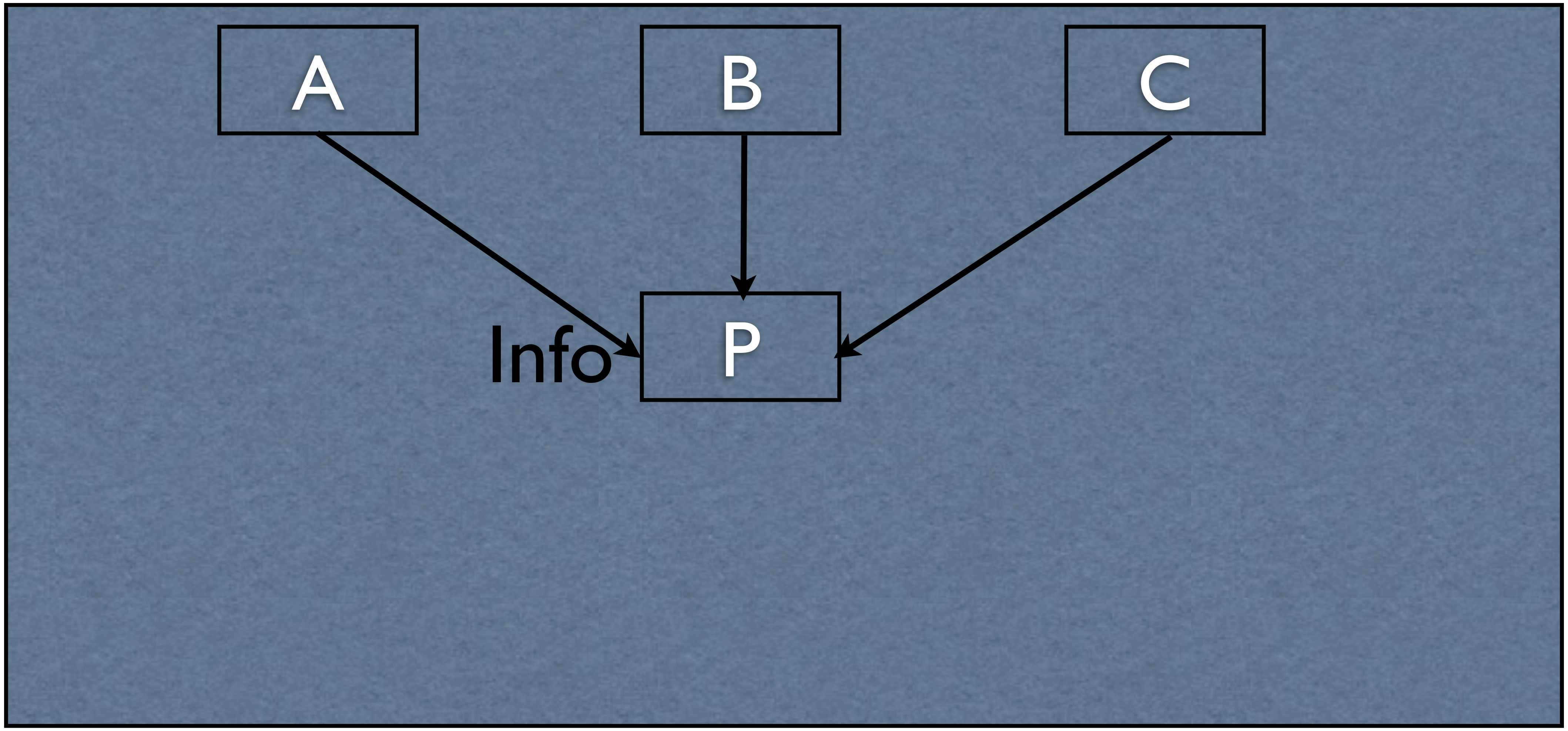
Break Table ..Cost

- Phase1: linear
- Phase2: nlogn
- Phase3: nlogn
    - we can enhance the last phase by constructing a hash if we have enough space.
    - Other suggestions to keep a linked list in holes and update pointers before moving objects.

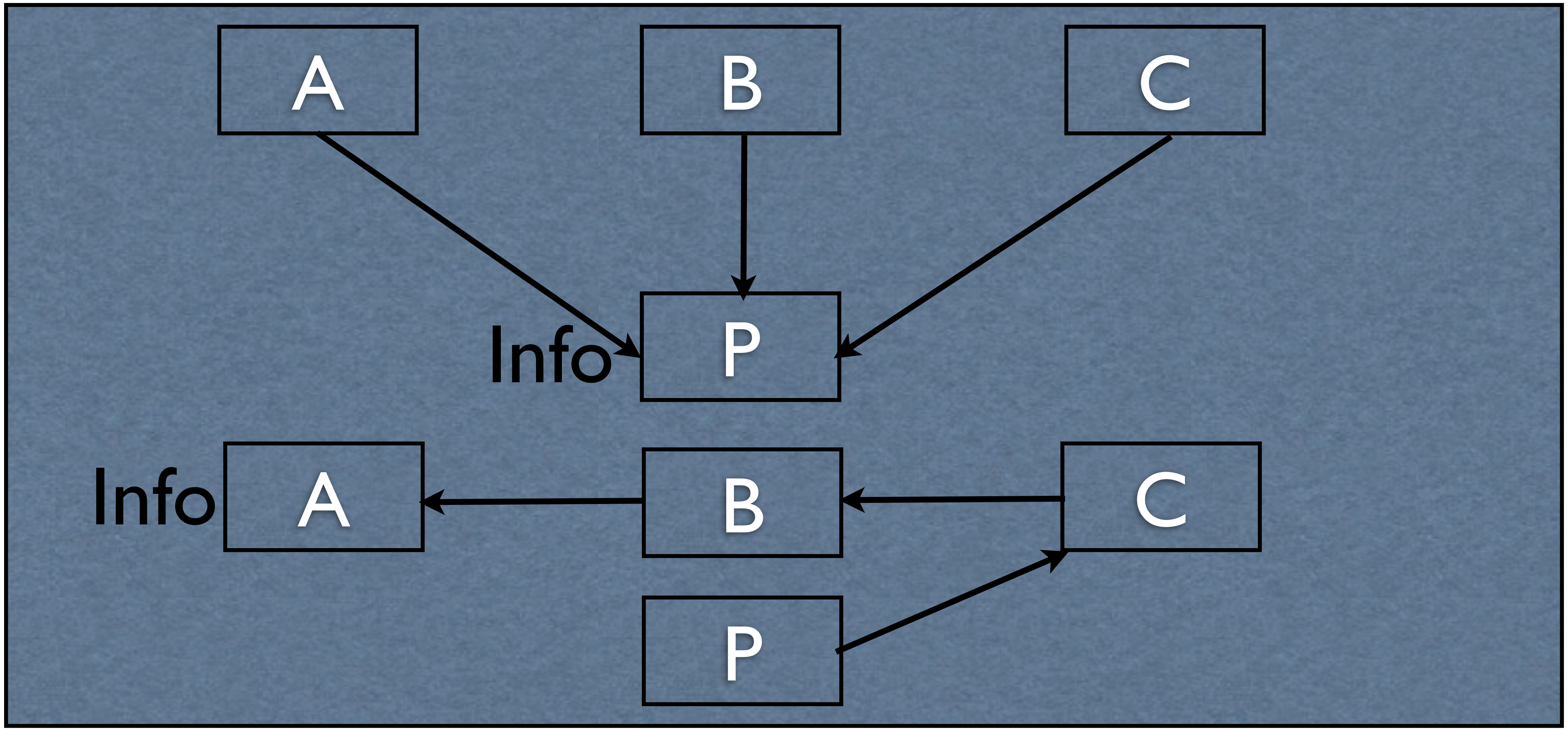
Problem .. revisited

- It is clear from the previous 2 algorithms that updating pointers is bottleneck.

Threading



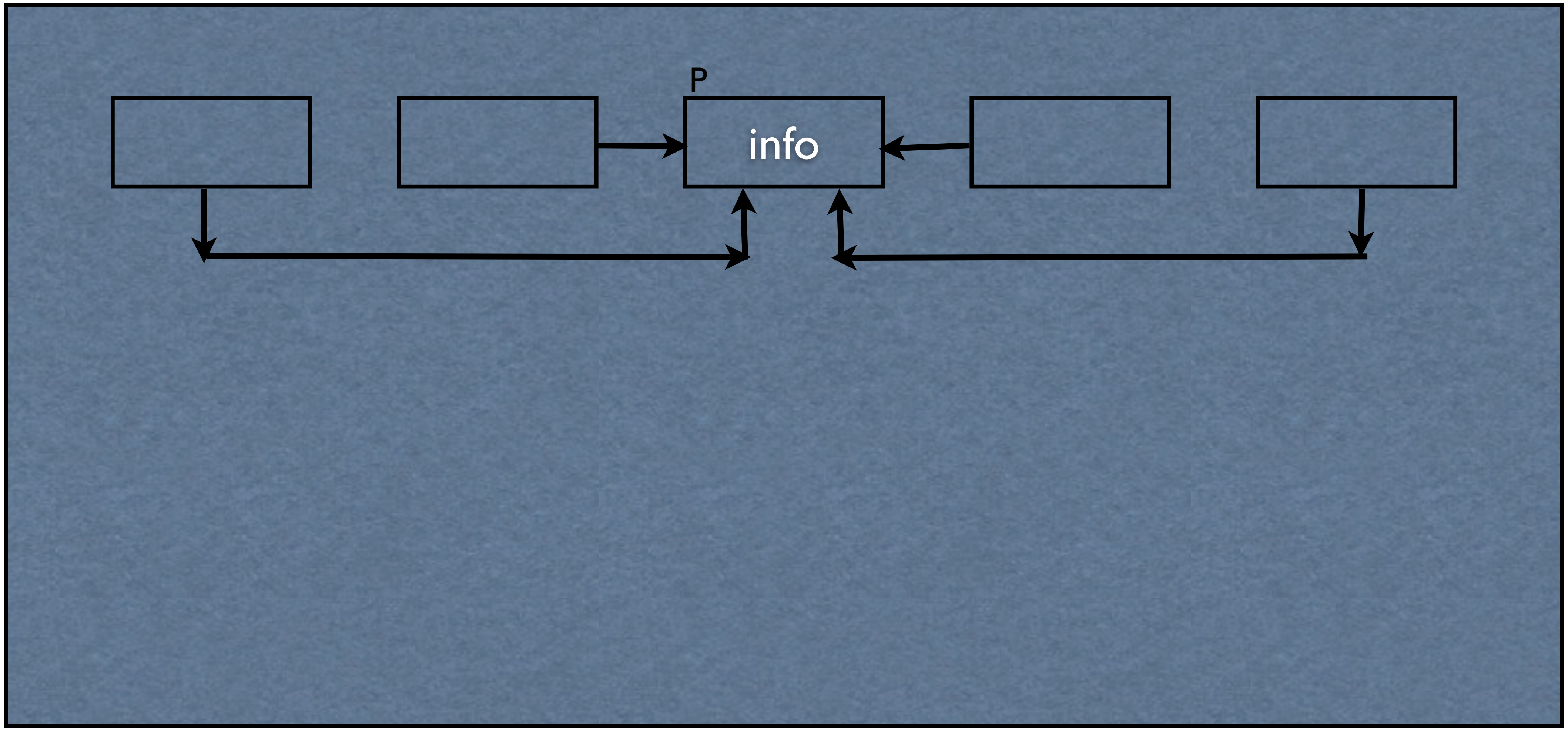
Threading



Threading

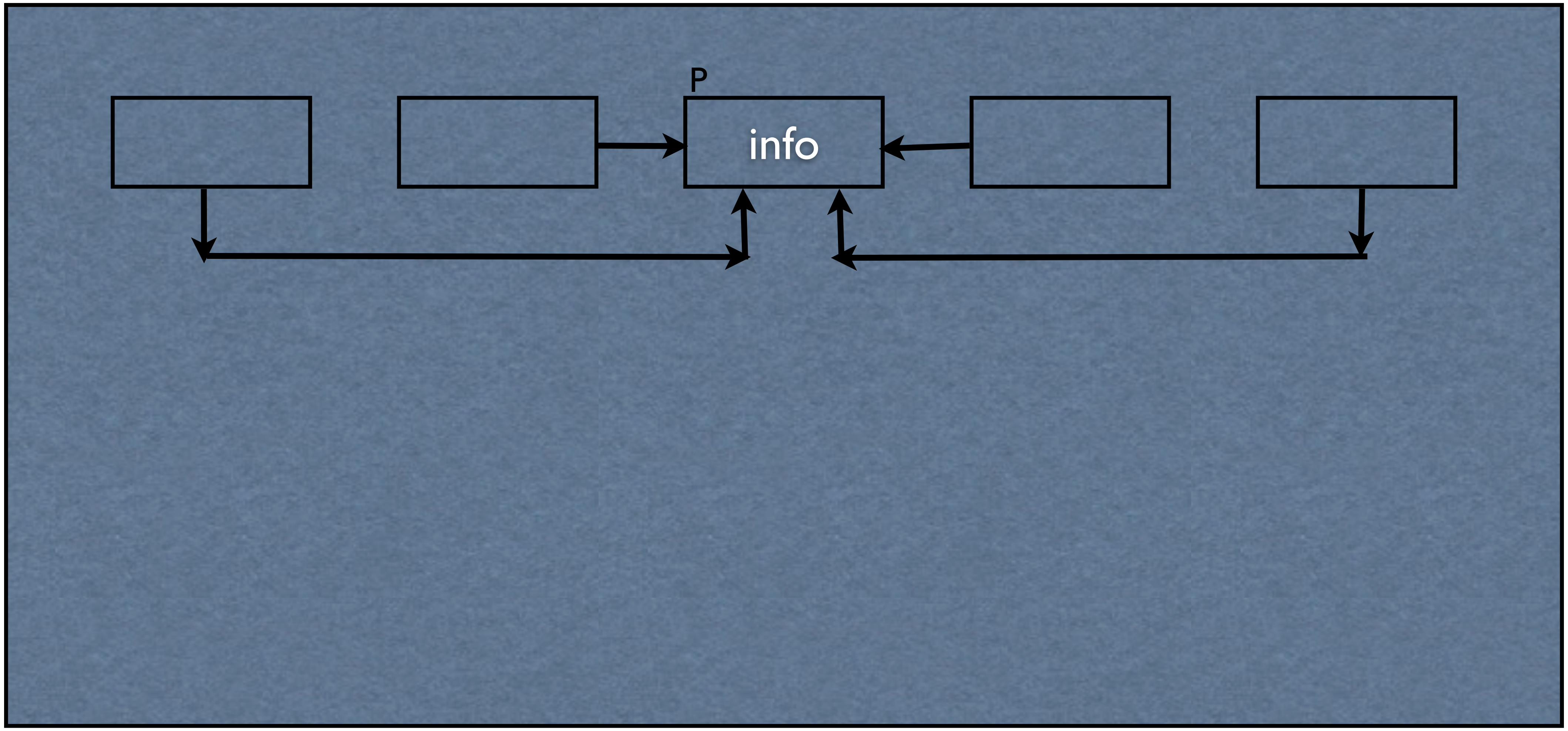
- After calculating the new address of P we can traverse the list and fix all the pointers to point to the new address of P.

Jonker Algorithm



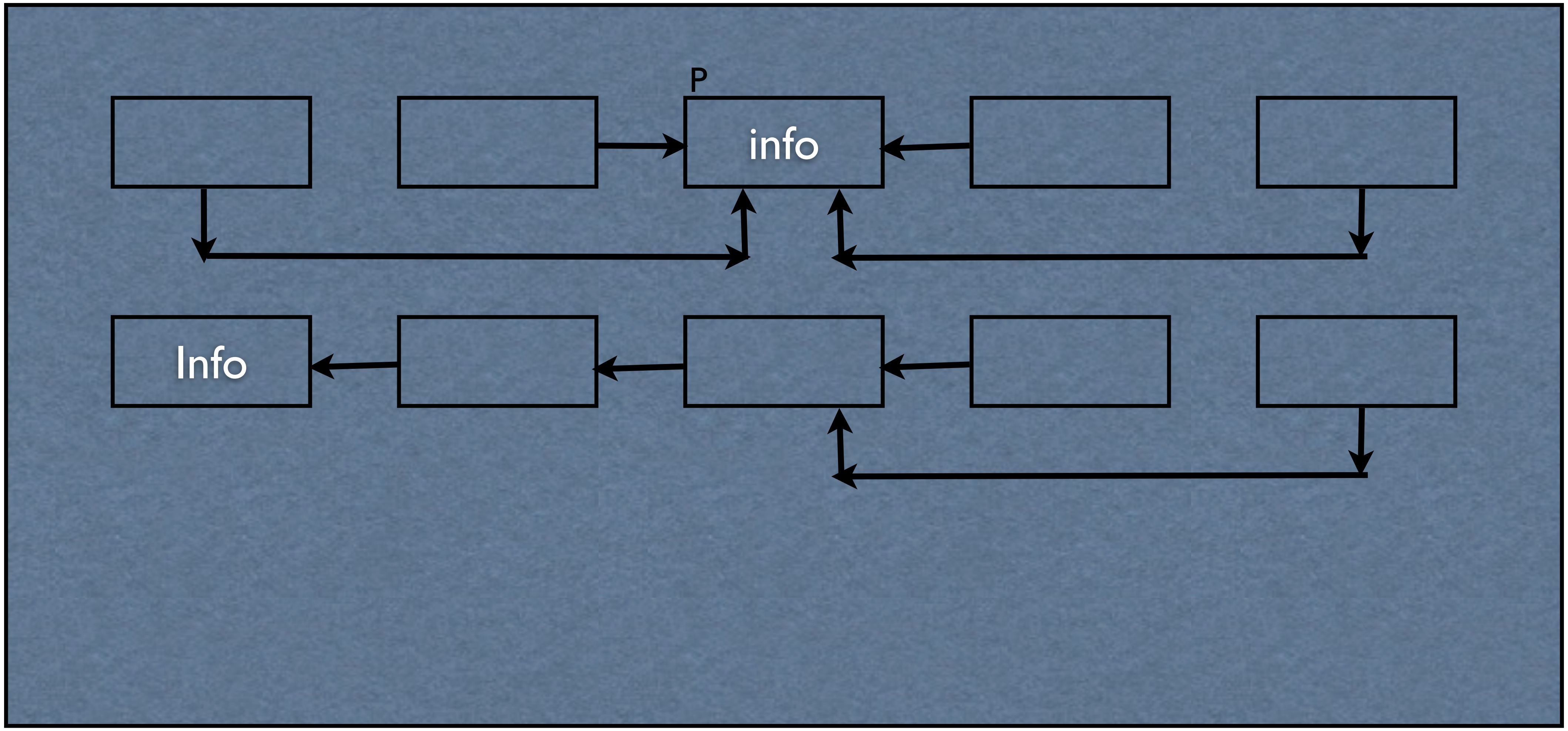
Jonker Algorithm

First Path



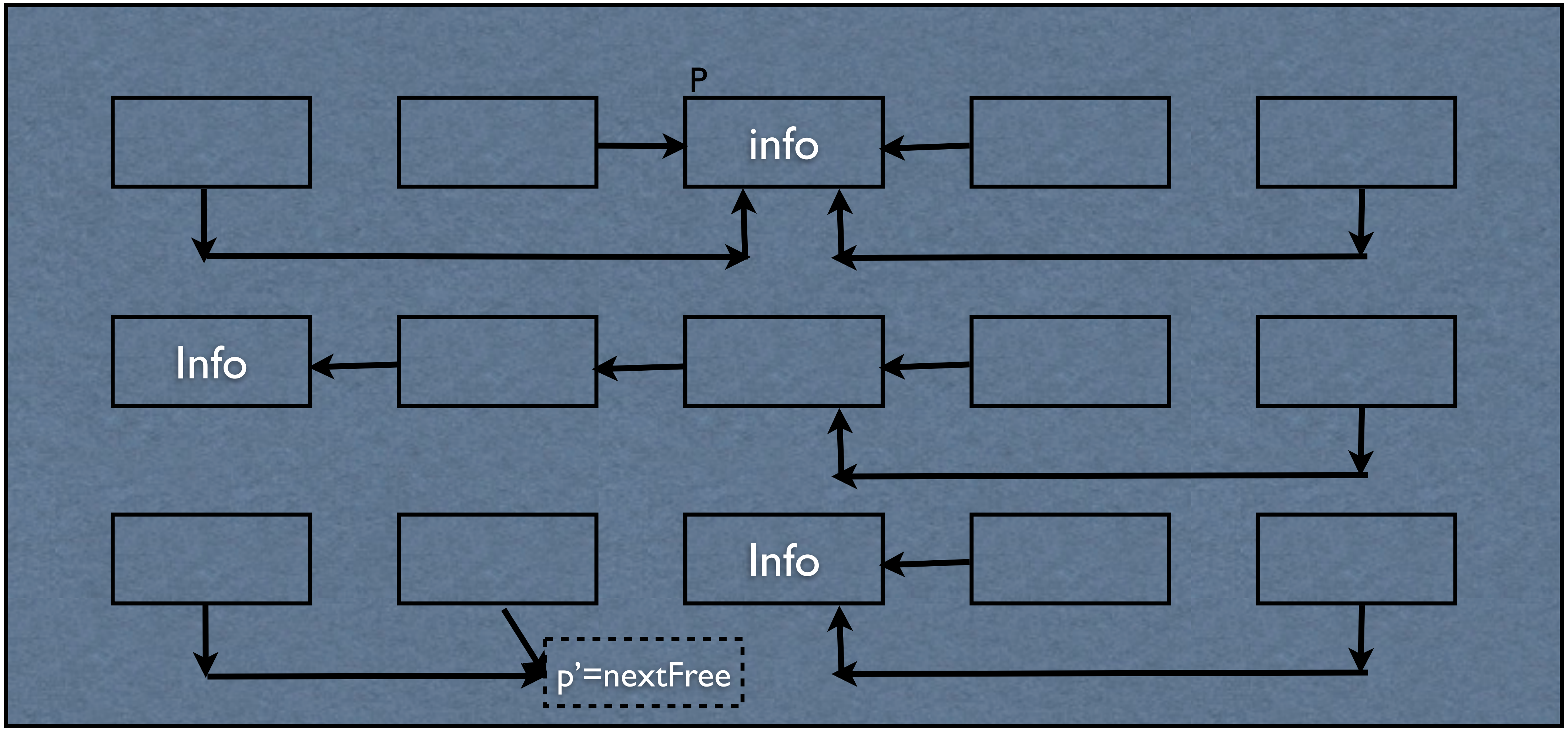
Jonker Algorithm

First Path



Jonker Algorithm

First Path



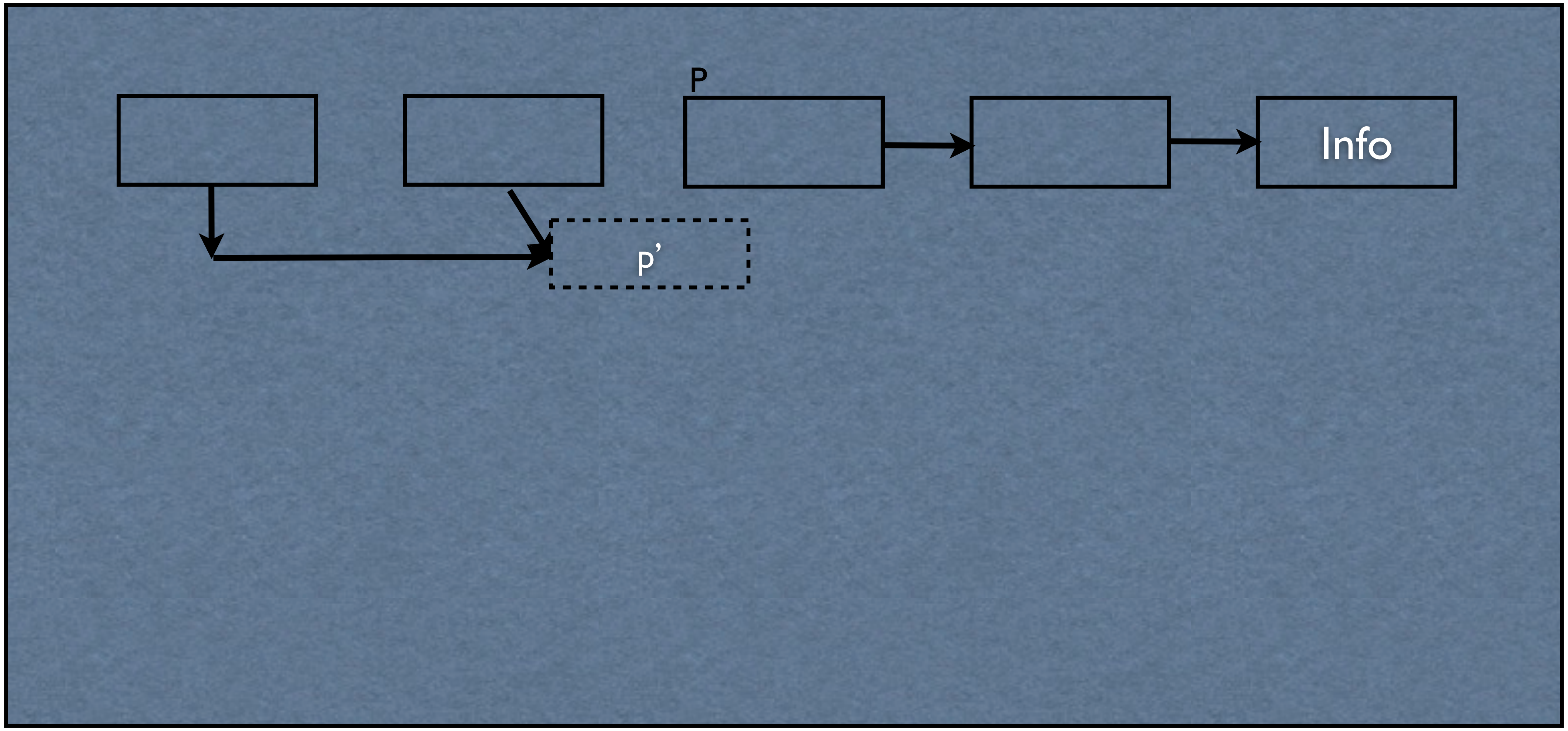
Jonker Algorithm

Jonker Algorithm

Second Path

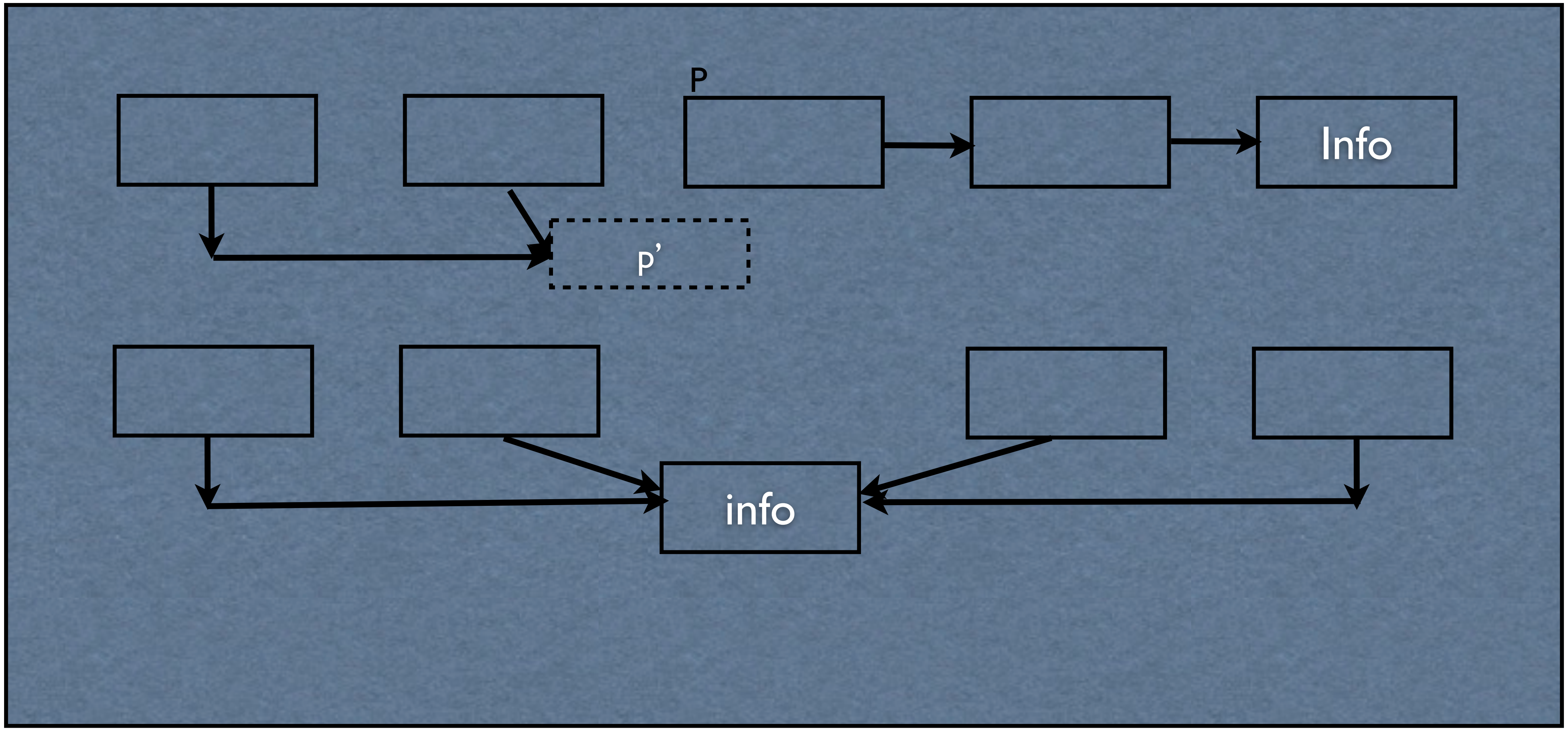
Jonker Algorithm

Second Path



Jonker Algorithm

Second Path



Analysis of Threaded

- Each object is touched three times.
- Space:
  - Jonker, no space required but each node has a pointer-sized header.
  - Morris
    - 2 tag bits per field, 0 inactive, 1 pointer, 2 swapped pointer, 3 non pointer.
- Could be improved by merging marking phase with first phase.

Threaded..Analysis

- Compact tables touch every object only twice.

Compaction Summary

- Suits smaller physical memory. Semi-Space requires double the memory space.
- For long lived objects, the heap becomes similar to “generational collector”.
- Improve locality.
- Other algorithms have only one path.

How to Compare

- Variable sized objects?
- Directions?
- Have to tag pointer data?
- Time and Space Performance.

Time Comparison

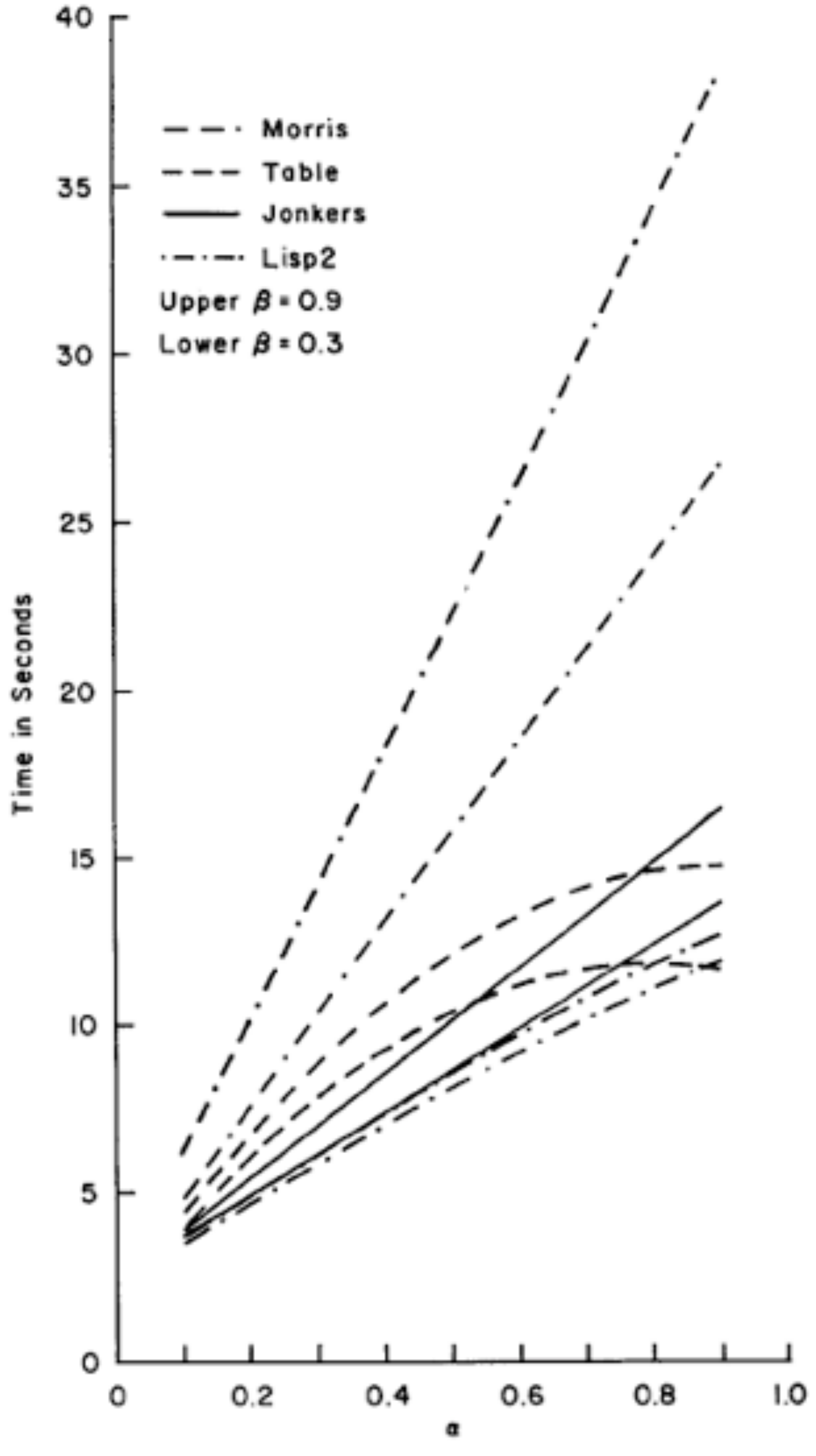
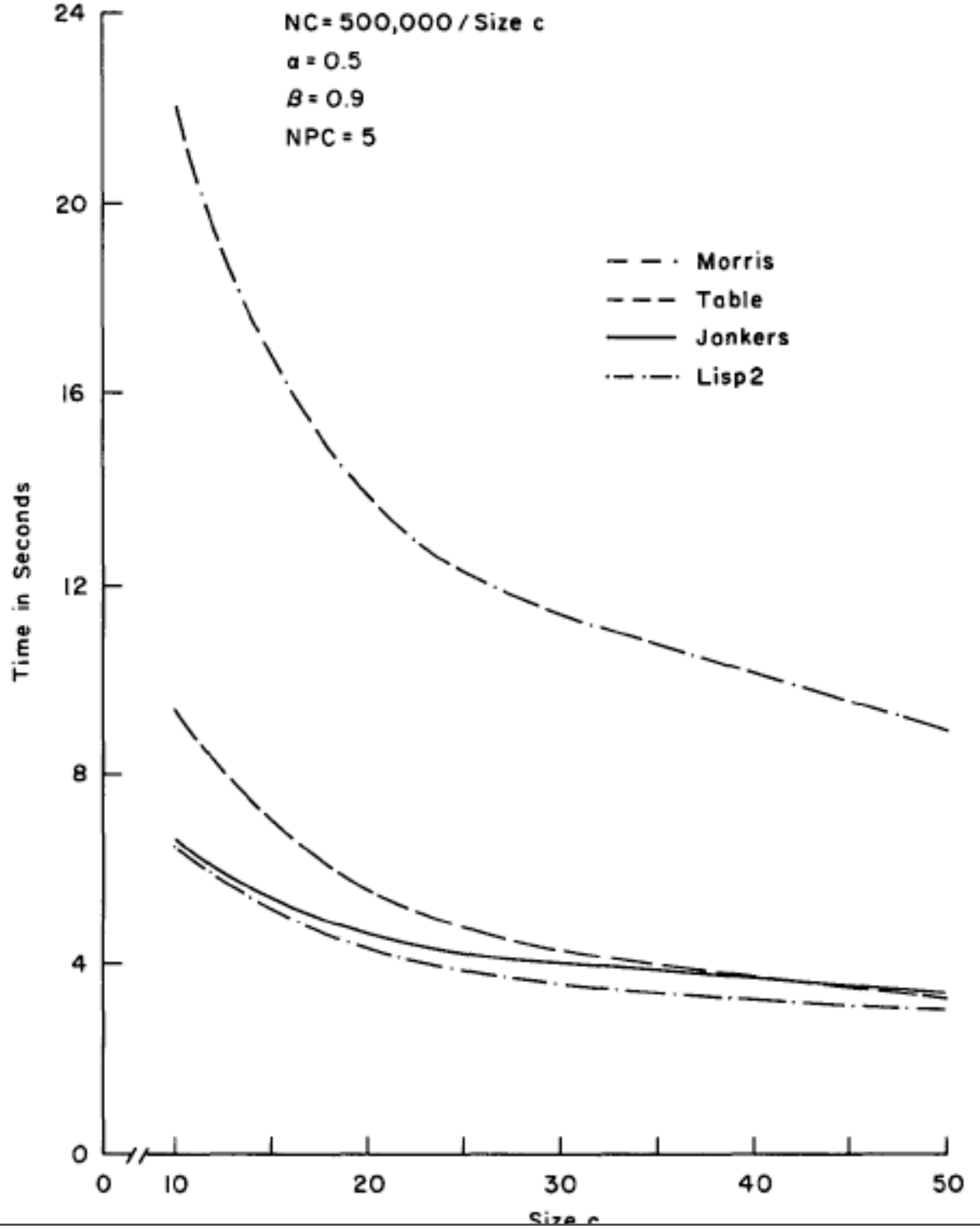


Fig. 8. Time comparisons for the four compactors.


Time in Seconds
Size c
NC = 500,000 / Size c
α = 0.5
β = 0.9
NPC = 5
Morris
Table
Jonkers
Lisp2