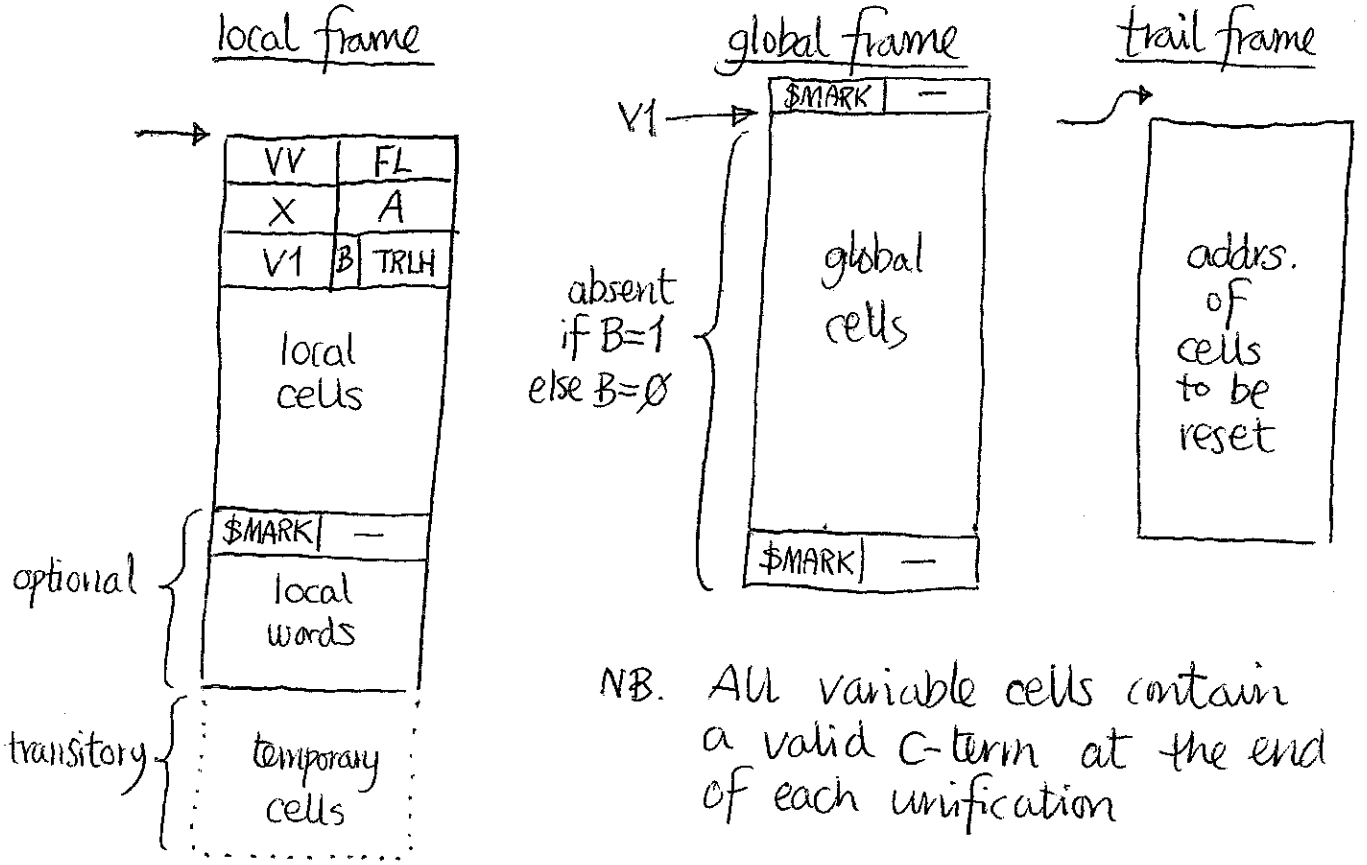
Layout of an Environment

Comprises a local frame, global frame, trail frame.



NB. All variable cells contain a valid C-term at the end of each unification

VV : choice point prior to this environment

FL : failure label for this environment, if any, undefined, otherwise.

X : parent environment

A : parent's arguments and continuation

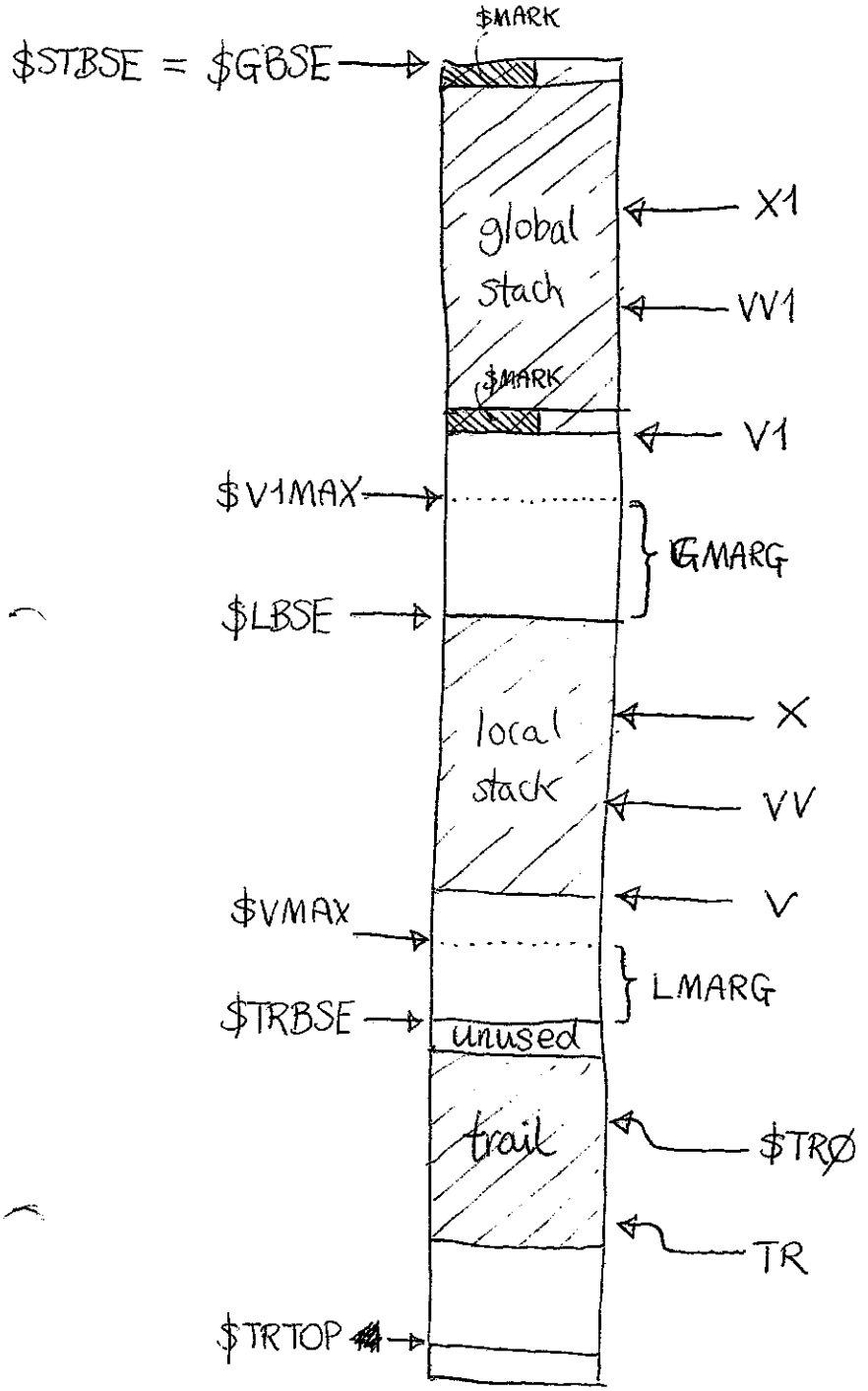
V1 : corresponding point on global stack = base of global frame if B=Ø.

B : bit indicating absence of global frame if B=1

TRLH : left half of TR prior to creation of this environment

$MARK = 777777_8

Layout of Memory



Interupts

$OFL : V > $VMAX

$OFL1 : V1 > $V1MAX

} All frames complete.
{X, X1} point at last {local, global} frames

pdl overflow from TR : TR = (Ø, $TRTOP)

} All frames complete except last ones.
{V, V1} point at last {local, global} frames.
Size of this last {local, global} frame ≤ {LMARG, GMARG}.

Remapping Memory

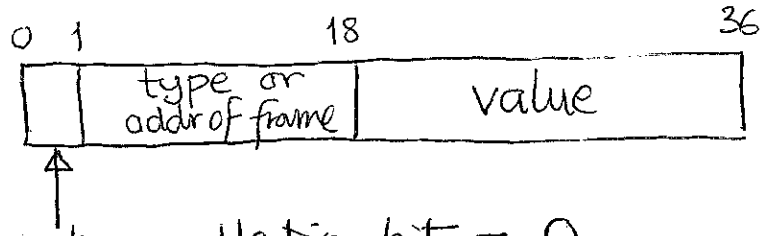
* Shifting Trail
  : Update TR, $TRØ, $TRB, $TRTOP, $VMAX.

* Changing Trail Quota
  : Update TR, $TRØ, $TRTOP,
  and each TRLH entry in local stack.

* Shifting Local Stack
  : Update X, VV, V, $LBSE, $V1MAX,
  and each {VV, X} entry in local stack,
  and each local cell containg a local ref,
  and each trail item containg a local ref.

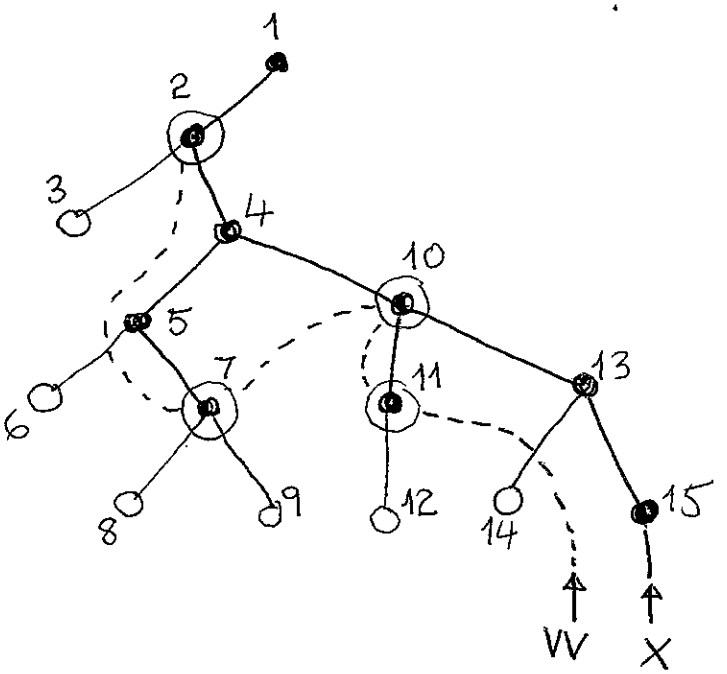
* Shifting Global Stack
  : Update X1, VV1, V1, $GBSE,
  and each V1 entry in local stack,
  and each cell containing a global ref
  or a molecule,
  and each trail item containg a global ref.

Format of a Constructed Term



except during garbage coll'n when global variable cells which are in use are marked with a 1.

How to trace back thru all local frames



Key

- backtrackpoint
- local & global frames
- global frame only

The parental (X) and backtrack (VV) links serve to trace all local frames (and corresponding active global frames). ie.

```
while VV ≥ $env∅ do
    (while X > VV do
        (trace X
         X := X.X)
     trace VV
     X := VV.X
     VV := VV.V)
```

Garbage Collection Algorithm

preliminaries: for each active global frame
mark each cell

trace&marking: for each local frame & corrg. active global frame
trace&mark each local cell
trace & mark each global cell

compute remapping: for each global frame
Compute displacement
(or $=\infty$. if to be dropped)

remap global addrs:
for each local frame
remap V1
remap each local cell
for each global frame
remap each global cell
for each trail item
remap reset ref (if to a global)
also remap VV1, X1, V1

Compaction:
compact global stack,
unmarking each cell in the process.

Remapping

global ref : scan backwards to $mark & subtract displacement

molecule : find* a variable in its skeleton, scan backwards from ~~corresponding~~ point in global stack to $mark & subtract displacement (if no vars, ie ground skeleton, do naught).

Trace & marking

Use a PDL {or / painting at standard PDLSPACE} set up ~~setup of~~ above local stack? (This PDL also for *)

Don't trace a local ref.

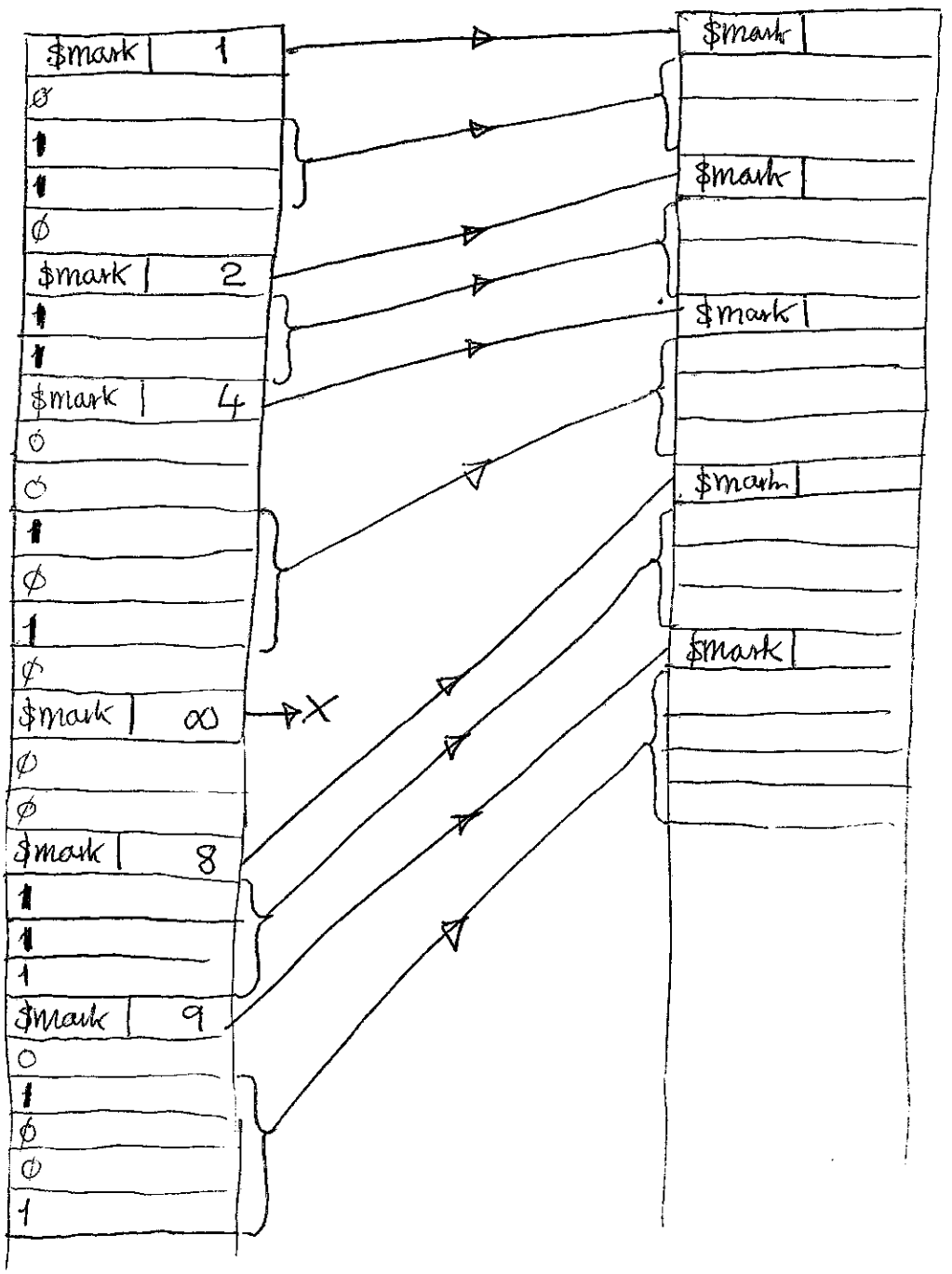
Don't trace if already marked.

Use the same device as in 'display' to avoid recursing on the final arg of a skel.

Compacting the Global Stack

After trace&mark-ing and computation of remap displacements

After compaction



Use of Fast Registers in Mark 2

Permanent Allocations

Octal	Decimal	Name	Use
11	9	VV1	Always: VV1 = VV·V1 (backtrack global frame)
12	10	X1	Caller's global frame.
13	11	VV	Backtrack local frame.
14	12	X	Caller's local frame.
15	13	V1	Origin for next global frame.
16	14	V	Origin for next local frame.
17	15	TR	Top of Trail push-down-list.

During Unification

Ø	FL	Failure label, but only when VV = V
1	T	} terms (input or constructed) passed as arguments to unification routines
2	B1	
3	B	
4	Y	address of a global frame
5	A	lh = X, rh = caller's arguments
6	R1	} temporary results
7	R2	
8	C	link register (return address for a run-time routine)

In the Body of a Clause

Ø	ACL	temporary accumulator
1–5	ACØ–AC4	accumulators
6	ACH	temporary accumulator
5	VAL	value register (of an evaluated expression)

Other locations affected by stack shifts

$TRB : base of trail

$GBSE (= $STBSE) ; base of ~~local~~ global stack.

$LBSE : base of local stack.

$VMAX : top of local stack − LMARG

$VIMAX : top of global stack − GMARG

where LMARG=50, GMARG=50 at present

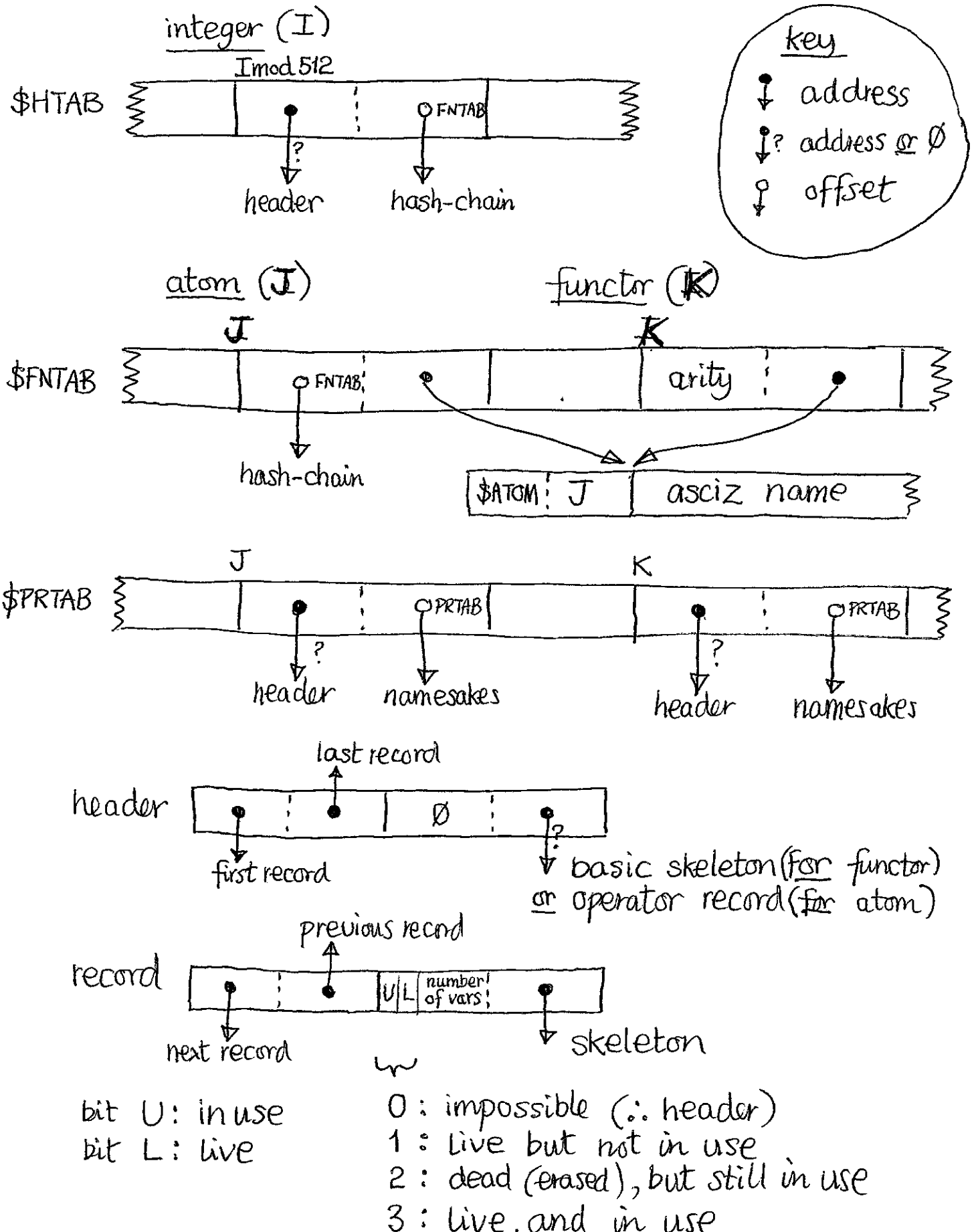
$TRØ : an earlier value of trail

$PDLØ : p.d.l. painter to push down list space.

$GRAT : (numerator, denominator) of global inc. ratio

$LRAT : (numerator, denominator) of local inc ratio

Information attached to integers, atoms, functors



Notes – Info. attached to integers, atoms, functors

- The "hash chain" links atoms having the same hash code (range 0–511), and is terminated by 777777_8.
- The "namesakes chain" links an atom with functors having the same name, and is terminated by a Ø. The atom comes first.
- The "header record" anchors a doubly-linked list of items recorded under a particular atom, functor or integer.
- The hash table, $HTAB, has a fixed size of 512 words.
- The functors and properties tables ($FNTAB, $PRTAB) are allocated "in tandem", in chunks scattered through the low segment.
- The basic skeleton for a functor Ø, arity N, is a skeleton corresponding to the term

$$\Phi(X1, X2, \ldots, XN).$$

System Database Access

tagged (Atom, Term)
tag (Atom, Term)
untag (Atom, Term)

Should preferably be an Atom.
No error messages.
Term must not be ':-'/2
Term ≠ Atom. or '$record'/1 or 2

User Database Access

recorded (Key, Term, Ref)
recorda (Key, Term, Ref)
recordz (Key, Term, Ref)
erase (Ref)
instance (Ref, Term)

If Key is an integer, special case.

Interpreted Program Access

clause (Head, Body) ~~Ref~~)
asserta (Clause) ~~Ref~~)
assertz (Clause) ~~Ref~~)
assert (Clause) ~~Ref~~)
retract (Clause)
supersede (Name, Arity)
reinstate (Name, Arity)

All goals must be specified.

Works both ways. {+??, ??+}

clause (Head, Body, Ref)
asserta (Clause, Ref)
assertz (Clause, Ref)
assert (Clause, Ref)
~~Erase (Ref)~~
~~Update (Ref)~~

Expanding Grammar Rules etc. {Would include macro expansions}

expand_term (Term1, Term2).