Dynamic Tables

Tobias Nipkow

March 17, 2025

Abstract

This article formalizes the amortized analysis of dynamic tables parameterized with their minimal and maximal load factors and the expansion and contraction factors.

A full description is found in a companion paper [1].

theory *Tables-real*
imports *Amortized-Complexity.Amortized-Framework0*
begin

fun Ψ :: *bool* $\Rightarrow$ *real* $\Rightarrow$ *real* $\Rightarrow$ *real* $\Rightarrow$ *real* $\Rightarrow$ *nat* $\Rightarrow$ *real* **where**
$\Psi\ b\ i\ d\ x_1\ x_2\ n = (\mathit{if}\ n \geq x_2\ \mathit{then}\ i{*}(n - x_2)\ \mathit{else}$
$\mathit{if}\ n \leq x_1 \wedge b\ \mathit{then}\ d{*}(x_1 - n)\ \mathit{else}\ 0)$

declare *of-nat-Suc*[*simp*] *of-nat-diff*[*simp*]

An automatic proof:

lemma *Psi-diff-Ins*:
$0 < i \Longrightarrow 0 < d \Longrightarrow \Psi\ b\ i\ d\ x_1\ x_2\ (Suc\ n) - \Psi\ b\ i\ d\ x_1\ x_2\ n \leq i$
⟨*proof*⟩

lemma assumes [*arith*]: $0 < i\ 0 \leq d$
shows $\Psi\ b\ i\ d\ x_1\ x_2\ (n{+}1) - \Psi\ b\ i\ d\ x_1\ x_2\ n \leq i$ (**is** *?D* $\leq$ -)
⟨*proof*⟩

lemma *Psi-diff-Del*: **assumes** [*arith*]: $0 < i\ 0 \leq d\ n{\neq}0$ **and** $x_1 \leq x_2$
shows $\Psi\ b\ i\ d\ x_1\ x_2\ (n{-}Suc\ 0) - \Psi\ b\ i\ d\ x_1\ x_2\ (n) \leq d$ (**is** *?D* $\leq$ -)
⟨*proof*⟩

locale *Table0* =
fixes *f1 f2 f1′ f2′ e c* :: *real*
assumes *e1*[*arith*]: $e > 1$
assumes *c1*[*arith*]: $c > 1$
assumes *f1*[*arith*]: $f1 > 0$
assumes *f1cf2*: $f1{*}c < f2$
assumes *f1f2e*: $f1 < f2/e$

assumes *f1'-def*: $f1' = min\ (f1*c)\ (f2/e)$
assumes *f2'-def*: $f2' = max\ (f1*c)\ (f2/e)$
begin

lemma *f2*[*arith*]: $0 < f2$
⟨*proof*⟩

lemma *f2'*[*arith*]: $0 < f2'$
⟨*proof*⟩

lemma *f2'-less-f2*: $f2' < f2$
⟨*proof*⟩

lemma *f1-less-f1'*: $f1 < f1'$
⟨*proof*⟩

lemma *f1'-gr0*[*arith*]: $f1' > 0$
⟨*proof*⟩

lemma *f1'-le-f2'*: $f1' \leq f2'$
⟨*proof*⟩

lemma *f1'c-le-f1*: $f1'/c \leq f1$
⟨*proof*⟩

lemma *f2-le-f2'e*: $f2 \leq f2'*e$
⟨*proof*⟩

lemma *f1f2'c*: $f1 \leq f2'/c$
⟨*proof*⟩

lemma *f1'ef2*: $f1' * e \leq f2$
⟨*proof*⟩

end

locale *Table* = *Table0* +
fixes *l0* :: *real*
assumes *l0f2e*: $l0 \geq 1/(f2 * (e-1))$
assumes *l0f1c*: $l0 \geq 1/(f1 * (c-1))$
assumes *f2f2'*: $l0 \geq 1/(f2 - f2')$
assumes *f1'f1*: $l0 \geq 1/((f1' - f1)*c)$
begin

definition $ai = f2/(f2-f2')$
definition $ad = f1/(f1'-f1)$

lemma *aigr0*[*arith*]: $ai > 1$
⟨*proof*⟩

lemma *adgr0*[*arith*]: *ad* > *0*
⟨*proof*⟩

lemma *l0-gr0*[*arith*]: *l0* > *0*
⟨*proof*⟩

lemma *f1-l0*: **assumes** *l0* ≤ *l*/*c* **shows** *f1*∗(*l*/*c*) ≤ *f1*∗*l* − *1*
⟨*proof*⟩

fun *nxt* :: op_{tb} ⇒ *nat*∗*real* ⇒ *nat*∗*real* **where**
nxt Ins (*n*,*l*) =
 (*n*+*1*, *if n*+*1* ≤ *f2*∗*l then l else e*∗*l*) |
nxt Del (*n*,*l*) =
 (*n*−*1*, *if f1*∗*l* ≤ *real*(*n*−*1*) *then l else if l0* ≤ *l*/*c then l*/*c else l*)

fun *T* :: op_{tb} ⇒ *nat*∗*real* ⇒ *real* **where**
T Ins (*n*,*l*) = (*if n*+*1* ≤ *f2*∗*l then 1 else n*+*1*) |
T Del (*n*,*l*) = (*if f1*∗*l* ≤ *real*(*n*−*1*) *then 1 else if l0* ≤ *l*/*c then n else 1*)

fun Φ :: *nat* ∗ *real* ⇒ *real* **where**
Φ (*n*,*l*) = (*if n* ≥ *f2*′∗*l then ai*∗(*n* − *f2*′∗*l*) *else*
 if n ≤ *f1*′∗*l* ∧ *l0* ≤ *l*/*c then ad*∗(*f1*′∗*l* − *n*) *else 0*)

lemma *Phi-Psi*: Φ (*n*,*l*) = Ψ (*l0* ≤ *l*/*c*) *ai ad* (*f1*′∗*l*) (*f2*′∗*l*) *n*
⟨*proof*⟩

fun *invar* **where**
invar(*n*,*l*) = (*l* ≥ *l0* ∧ (*l*/*c* ≥ *l0* ⟶ *f1*∗*l* ≤ *n*) ∧ *n* ≤ *f2*∗*l*)

abbreviation *U* ≡ λ*f* -. *case f of Ins* ⇒ *ai*+*1* | *Del* ⇒ *ad*+*1*

interpretation *tb*: *Amortized*
 where *init* = (*0*,*l0*) **and** *nxt* = *nxt*
 and *inv* = *invar*
 and *T* = *T* **and** Φ = Φ
 and *U* = *U*
⟨*proof*⟩

end

locale *Optimal* =
fixes *f2 c e* :: *real* **and** *l0* :: *nat*
assumes *e1*[*arith*]: *e* > *1*
assumes *c1*[*arith*]: *c* > *1*
assumes [*arith*]: *f2* > *0*
assumes *l0*: (*e*∗*c*)/(*f2*∗(*min e c* − *1*)) ≤ *l0*
begin

lemma *l0e*: $(e*c)/(f2*(e-1)) \leq l0$
⟨*proof*⟩

lemma *l0c*: $(e*c)/(f2*(c-1)) \leq l0$
⟨*proof*⟩

interpretation *Table*
where $f1=f2/(e*c)$ **and** $f2=f2$ **and** $e=e$ **and** $c=c$ **and** $f1'=f2/e$ **and** $f2'=f2/e$
and $l0=l0$
⟨*proof*⟩

lemma $ai = e/(e-1)$
⟨*proof*⟩

lemma $ad = 1/(c-1)$
⟨*proof*⟩

end

interpretation *I1*: *Optimal* **where** $e=2$ **and** $c=2$ **and** $f2=1$ **and** $l0=4$
⟨*proof*⟩

interpretation *I2*: *Optimal* **where** $e=2$ **and** $c=2$ **and** $f2=3/4$ **and** $l0=6$
⟨*proof*⟩

interpretation *I3*: *Optimal* **where** $e=2$ **and** $c=2$ **and** $f2=0.8$ **and** $l0=5$
⟨*proof*⟩

interpretation *I4*: *Optimal* **where** $e=3$ **and** $c=3$ **and** $f2=0.9$ **and** $l0=5$
⟨*proof*⟩

interpretation *I5*: *Optimal* **where** $e=4$ **and** $c=4$ **and** $f2=1$ **and** $l0=6$
⟨*proof*⟩

interpretation *I6*: *Optimal* **where** $e=2.5$ **and** $c=2.5$ **and** $f2=1$ **and** $l0=5$
⟨*proof*⟩

interpretation *I7*: *Optimal* **where** $f2=1$ **and** $c=3/2$ **and** $e=2$ **and** $l0=6$
⟨*proof*⟩

interpretation *I8*: *Optimal* **where** $f2=1$ **and** $e=3/2$ **and** $c=2$ **and** $l0=6$
⟨*proof*⟩

end
theory *Tables-nat*
imports *Tables-real*
begin

declare *le-of-int-ceiling*[*simp*]

locale *TableInv* = *Table0 f1 f2 f1′ f2′ e c* **for** *f1 f2 f1′ f2′ e c* :: *real* +
fixes *l0* :: *nat*
assumes *l0f2e*: $l0 \geq 1/(f2 * (e-1))$
assumes *l0f1c*: $l0 \geq 1/(f1 * (c-1))$

assumes *l0f2f1e*: $l0 \geq f1/(f2 - f1*e)$
assumes *l0f2f1c*: $l0 \geq f2/(f2 - f1*c)$
begin

lemma *l0-gr0*[*arith*]: $l0 > 0$
⟨*proof*⟩

lemma *f1-l0*: **assumes** $l0 \leq l/c$ **shows** $f1*(l/c) \leq f1*l - 1$
⟨*proof*⟩

fun *nxt* :: $op_{tb} \Rightarrow nat*nat \Rightarrow nat*nat$ **where**
nxt Ins (*n*,*l*) =
 ($n+1$, *if* $n+1 \leq f2*l$ *then l else nat*$\lceil e*l \rceil$) |
nxt Del (*n*,*l*) =
 ($n-1$, *if* $f1*l \leq real(n-1)$ *then l else if* $l0 \leq \lfloor l/c \rfloor$ *then nat*$\lfloor l/c \rfloor$ *else l*)

fun *T* :: $op_{tb} \Rightarrow nat*nat \Rightarrow real$ **where**
T Ins (*n*,*l*) = (*if* $n+1 \leq f2*l$ *then 1 else* $n+1$) |
T Del (*n*,*l*) = (*if* $f1*l \leq real(n-1)$ *then 1 else if* $l0 \leq \lfloor l/c \rfloor$ *then n else 1*)

fun *invar* :: $nat * nat \Rightarrow bool$ **where**
$invar(n,l) = (l \geq l0 \wedge (\lfloor l/c \rfloor \geq l0 \longrightarrow f1*l \leq n) \wedge n \leq f2*l)$

lemma *invar-init*: *invar* (*0*,*l0*)
⟨*proof*⟩

lemma *invar-pres*: **assumes** *invar s* **shows** *invar*(*nxt f s*)
⟨*proof*⟩

end

locale *Table1* = *TableInv* +
assumes *f2f2′*: $l0 \geq 1/(f2 - f2')$
assumes *f1′f1*: $l0 \geq 1/((f1' - f1)*c)$
begin

definition $ai = f2/(f2-f2')$
definition $ad = f1/(f1'-f1)$

lemma *aigr0*[*arith*]: $ai > 1$

⟨*proof*⟩

lemma *adgr0*[*arith*]: *ad* > *0*
⟨*proof*⟩

lemma *f1′ad*[*arith*]: *f1′*∗*ad* > *0*
⟨*proof*⟩

lemma *f2′ai*[*arith*]: *f2′*∗*ai* > *0*
⟨*proof*⟩

fun Φ :: *nat* ∗ *nat* ⇒ *real* **where**
Φ (*n*,*l*) = (*if n* ≥ *f2′*∗*l then ai*∗(*n* − *f2′*∗*l*) *else*
  if n ≤ *f1′*∗*l* ∧ *l0* ≤ ⌊*l*/*c*⌋ *then ad*∗(*f1′*∗*l* − *n*) *else 0*)

lemma *Phi-Psi*: Φ (*n*,*l*) = Ψ (*l0* ≤ ⌊*l*/*c*⌋) *ai ad* (*f1′*∗*l*) (*f2′*∗*l*) *n*
⟨*proof*⟩

abbreviation *U* ≡ λ*f* -. *case f of Ins* ⇒ *ai*+*1* + *f1′*∗*ad* | *Del* ⇒ *ad*+*1* + *f2′*∗*ai*

interpretation *tb*: *Amortized*
  where *init* = (*0*,*l0*) **and** *nxt* = *nxt*
  and *inv* = *invar*
  and *T* = *T* **and** Φ = Φ
  and *U* = *U*
⟨*proof*⟩

end

locale *Table2-f1f2″* = *TableInv* +
fixes *f1″ f2″* :: *real*

locale *Table2* = *Table2-f1f2″* +
assumes *f2f2″*: (*f2* − *f2″*)∗*l0* ≥ *1*
assumes *f1″f1*: (*f1″* − *f1*)∗*c*∗*l0* ≥ *1*

assumes *f1-less-f1″*: *f1* < *f1″*
assumes *f1″-less-f1′*: *f1″* < *f1′*
assumes *f2′-less-f2″*: *f2′* < *f2″*
assumes *f2″-less-f2*: *f2″* < *f2*
assumes *f1″-f1′*: *l* ≥ *real l0* ⟹ *f1″* ∗ (*l*+*1*) ≤ *f1′*∗*l*
assumes *f2′-f2″*: *l* ≥ *real l0* ⟹ *f2′* ∗ *l* ≤ *f2″* ∗ (*l*−*1*)
begin

definition *ai* = *f2* / (*f2* − *f2″*)
definition *ad* = *f1* / (*f1″* − *f1*)

lemma *f1″-gr0*[*arith*]: *f1″* > *0*
⟨*proof*⟩

lemma *f2''-gr0*[*arith*]: $f2'' > 0$
⟨*proof*⟩

lemma *aigr0*[*arith*]: $ai > 0$
⟨*proof*⟩

lemma *adgr0*[*arith*]: $ad > 0$
⟨*proof*⟩

fun Φ :: *nat* $*$ *nat* $\Rightarrow$ *real* **where**
$\Phi(n,l) = (\mathit{if}\ n \geq f2''{*}l\ \mathit{then}\ ai{*}(n - f2''{*}l)\ \mathit{else}$
$\quad \mathit{if}\ n \leq f1''{*}l \wedge l0 \leq \lfloor l/c \rfloor\ \mathit{then}\ ad{*}(f1''{*}l - n)\ \mathit{else}\ 0)$

lemma *Phi-Psi*: $\Phi\ (n,l) = \Psi\ (l0 \leq \lfloor l/c \rfloor)\ ai\ ad\ (f1''{*}l)\ (f2''{*}l)\ n$
⟨*proof*⟩

abbreviation $U \equiv \lambda f\ \text{-}.\ \mathit{case}\ f\ \mathit{of}\ Ins \Rightarrow ai{+}1 \mid Del \Rightarrow ad{+}1$

interpretation *tb*: *Amortized*
  where $init = (0,l0)$ **and** $nxt = nxt$
  and $inv = invar$
  and $T = T$ **and** $\Phi = \Phi$
  and $U = U$
⟨*proof*⟩

end

locale *Table3* = *Table2-f1f2''* +
assumes *f1''-def*: $f1'' = (f1'{::}real){*}l0/(l0{+}1)$
assumes *f2''-def*: $f2'' = (f2'{::}real){*}l0/(l0{-}1)$

assumes *l0-f2f2'*: $l0 \geq (f2{+}1)/(f2{-}f2')$
assumes *l0-f1f1'*: $l0 \geq (f1'{*}c{+}1)/((f1'{-}f1){*}c)$

assumes *l0-f1-f1'*: $l0 > f1/((f1'{-}f1))$
assumes *l0-f2-f2'*: $l0 > f2/(f2{-}f2')$
begin

lemma *l0-gr1*: $l0 > 1$
⟨*proof*⟩

lemma *f1''-less-f1'*: $f1'' < f1'$
⟨*proof*⟩

lemma *f1-less-f1''*: $f1 < f1''$

⟨*proof*⟩

lemma $f2'$-less-$f2''$: $f2' < f2''$
⟨*proof*⟩

lemma $f2''$-less-$f2$: $f2'' < f2$
⟨*proof*⟩

lemma $f2f2''$: $(f2 - f2'')*l0 \geq 1$
⟨*proof*⟩

lemma $f1''f1$: $(f1'' - f1)*c*l0 \geq 1$
⟨*proof*⟩

lemma $f1''$-$f1'$: **assumes** $l \geq real\ l0$ **shows** $f1''*(l+1) \leq f1' * l$
⟨*proof*⟩

lemma $f2'$-$f2''$: **assumes** $l \geq real\ l0$ **shows** $f2' * l \leq f2'' * (l-1)$
⟨*proof*⟩

sublocale *Table2*
⟨*proof*⟩

end

end

References

[1] T. Nipkow. Parameterized dynamic tables. http://www.in.tum.de/~nipkow/pubs/, 2015.