Dynamic Tables

Tobias Nipkow

April 18, 2024

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**
Ψ *b i d* x_1 x_2 *n* = (*if* $n \geq x_2$ *then* $i*(n - x_2)$ *else*
  if $n \leq x_1 \wedge b$ *then* $d*(x_1 - n)$ *else 0*)

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

An automatic proof:

lemma *Psi-diff-Ins*:
  $0 < i \implies 0 < d \implies \Psi$ *b i d* x_1 x_2 *(Suc n)* $- \Psi$ *b i d* x_1 x_2 *n* $\leq i$
by (*simp add*: *add-mono algebra-simps*)

lemma assumes [*arith*]: $0 < i$ $0 \leq d$
shows Ψ *b i d* x_1 x_2 *(n+1)* $- \Psi$ *b i d* x_1 x_2 *n* $\leq i$ (**is** $?D \leq$ -)
proof *cases*
  assume $n \geq x_2$
  hence $?D = i*(n+1-x_2) - i*(n-x_2)$ **by** (*simp*)
  also have ... = *i* **by** (*simp add*: *algebra-simps*)
  finally show *?thesis* **by** *simp*
next
  assume [*arith*]: $\neg n \geq x_2$
  show *?thesis*
  proof *cases*
    assume *∗*[*arith*]: $n \leq x_1 \wedge b$
    show *?thesis*
    proof *cases*
      assume $n+1 \geq x_2$

      hence *?D = i∗(n+1−x_2) − d∗(x_1−n)* **using** ∗ **by** (*simp*)
      also have *. . . = i + i∗(n−x_2) + −(d∗(x_1−n))*
        by (*simp add: algebra-simps*)
      also have *i∗(n−x_2) ≤ 0* **by** (*simp add: mult-le-0-iff*)
      also have *−(d∗(x_1−n)) ≤ 0* **using** ∗ **by** (*simp*)
      finally show *?thesis* **by** *simp*
    next
      assume [*arith*]: ¬ *n+1 ≥ x_2*
      thus *?thesis*
      proof *cases*
        assume *n+1 ≥ x_1*
        hence *?D = −(d∗(x_1−n))* **using** ∗ **by** (*simp*)
        also have *−(d∗(x_1−n)) ≤ 0* **using** ∗ **by** (*simp*)
        finally have *?D ≤ 0* **by** *simp*
        then show *?thesis* **by** *simp*
      next
        assume ¬ *n+1 ≥ x_1*
        hence *?D = d∗(x_1−(n+1)) − d∗(x_1−n)* **using** ∗ **by** (*simp*)
        also have *. . . = −d* **by** (*simp add: algebra-simps*)
        finally show *?thesis* **by** (*simp*)
      qed
    qed
  next
    assume ∗: ¬ *(n ≤ x_1 ∧ b)*
    show *?thesis*
    proof *cases*
      assume *n+1 ≥ x_2*
      hence *?D = i∗(n+1−x_2)* **using** ∗ **by** (*auto*)
      also have *. . . ≤ i* **by**(*simp add: algebra-simps*)
      finally show *?thesis* **by** *simp*
    next
      assume ¬ *n+1 ≥ x_2*
      hence *?D = 0* **using** ∗ **by** (*auto*)
      thus *?thesis* **by** *simp*
    qed
  qed
qed

lemma *Psi-diff-Del*: **assumes** [*arith*]: *0 < i 0 ≤ d n≠0* **and** *x_1 ≤ x_2*
shows Ψ *b i d x_1 x_2 (n−Suc 0)* − Ψ *b i d x_1 x_2 (n) ≤ d* (**is** *?D ≤ -*)
proof *cases*
  assume *real n − 1 ≥ x_2*
  hence *?D = i∗(n−1−x_2) − i∗(n−x_2)* **by**(*simp*)
  also have *. . . = − i* **by** (*simp add: algebra-simps*)
  finally show *?thesis* **by** *simp*
next
  assume [*arith*]: ¬ *real n − 1 ≥ x_2*
  show *?thesis*
  proof *cases*

assume ∗: $n{-}1 \leq x_1 \land b$
show *?thesis*
proof *cases*
  assume [*arith*]: $n \geq x_2$
  hence *f1*: $x_1 \leq n$ **using** ‹$x_1 \leq x_2$› **by** *linarith*
  have *?D* $= d{*}(x_1{-}(n{-}1)) - i{*}(n{-}x_2)$ **using** ∗ **by** (*simp*)
  also have $\ldots = d + d{*}(x_1{-}n) + -(i{*}(n{-}x_2))$
    by (*simp add: algebra-simps*)
  also have $-(i{*}(n{-}x_2)) \leq 0$ **by** (*simp add: mult-le-0-iff*)
  also have $d{*}(x_1{-}n) \leq 0$ **using** *f1* **by** (*simp add: mult-le-0-iff*)
  finally show *?thesis* **by** *simp*
next
  assume [*arith*]: ¬ $n \geq x_2$
  thus *?thesis*
  proof *cases*
    assume [*arith*]: $n > x_1$
    hence *?D* $= d{*}(x_1{-}(n{-}1))$ **using** ∗ **by** (*simp*)
    also have $\ldots = d + -(d{*}(n{-}x_1))$ **by** (*simp add: algebra-simps*)
    also have $-(d{*}(n{-}x_1)) \leq 0$ **by** (*simp add: mult-le-0-iff*)
    finally show *?thesis* **by** *simp*
  next
    assume ¬ $n > x_1$
    hence *?D* $= d{*}(x_1{-}(n{-}1)) - d{*}(x_1{-}n)$ **using** ∗ **by** (*simp*)
    also have $\ldots = d$ **by** (*simp add: algebra-simps*)
    finally show *?thesis* **by** (*simp*)
  qed
qed
next
  assume ∗: ¬ $(n{-}1 \leq x_1 \land b)$
  show *?thesis*
  proof *cases*
    assume *n*: $n \geq x_2$
    hence *?D* $= -(i{*}(n{-}x_2))$ **using** ∗ **by**(*auto*)
    also have $-(i{*}(n{-}x_2)) \leq 0$ **using** *n* **by**(*simp*)
    finally show *?thesis* **by** *simp*
  next
    assume ¬ $n \geq x_2$
    hence *?D* $= 0$ **using** ∗ **by** (*auto*)
    thus *?thesis* **by** *simp*
  qed
qed
qed

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*: $f1{*}c < f2$

3

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*
using *f1f2e zero-less-divide-iff*[*of f2 e*] **by** *simp*

lemma *f2'*[*arith*]: *0 < f2'*
by (*simp add*: *f2'-def max-def*)

lemma *f2'-less-f2*: *f2' < f2*
using *f1cf2* **by**(*auto simp add*: *f2'-def field-simps*)

lemma *f1-less-f1'*: *f1 < f1'*
using *f1f2e* **by**(*auto simp add*: *f1'-def field-simps*)

lemma *f1'-gr0*[*arith*]: *f1' > 0*
using *f1-less-f1'* **by** *linarith*

lemma *f1'-le-f2'*: *f1' ≤ f2'*
by(*auto simp add*: *f1'-def f2'-def algebra-simps*)

lemma *f1'c-le-f1*: *f1'/c ≤ f1*
by(*simp add*: *f1'-def min-def field-simps*)

lemma *f2-le-f2'e*: *f2 ≤ f2'∗e*
by(*simp add*: *f2'-def max-def field-simps*)

lemma *f1f2'c*: *f1 ≤ f2'/c*
using *f1f2e* **by**(*auto simp add*: *f2'-def field-simps*)

lemma *f1'ef2*: *f1' ∗ e ≤ f2*
using *f1cf2* **by**(*auto simp add*: *f1'-def field-simps min-def*)

end

locale *Table = Table0 +*
fixes *l0* :: *real*
assumes *l0f2e*: *l0 ≥ 1/(f2 ∗ (e−1))*
assumes *l0f1c*: *l0 ≥ 1/(f1 ∗ (c−1))*
assumes *f2f2'*: *l0 ≥ 1/(f2 − f2')*
assumes *f1'f1*: *l0 ≥ 1/((f1' − f1)∗c)*
begin

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

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

using *f2′-less-f2* **by**(*simp add*: *ai-def field-simps*)

lemma *adgr0*[*arith*]: *ad > 0*
using *f1-less-f1′* **by**(*simp add*: *ad-def field-simps*)

lemma *l0-gr0*[*arith*]: *l0 > 0*
proof −
  have $0 < 1/(f2*(e-1))$ **by**(*simp*)
  also note *l0f2e*
  finally show *?thesis* .
qed

lemma *f1-l0*: **assumes** $l0 \leq l/c$ **shows** $f1*(l/c) \leq f1*l - 1$
proof −
  have $1 = f1*((c-1)/c)*(c*(1/(f1*(c-1))))$ **by**(*simp add*: *field-simps*)
  also note *l0f1c*
  also note *assms*(*1*)
  finally show *?thesis* **by**(*simp add*: *divide-le-cancel*) (*simp add*: *field-simps*)
qed

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

fun *T* :: $op_{tb} \Rightarrow nat*real \Rightarrow real$ **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* $\leq$ *l/c then n else 1*)

fun Φ :: $nat * real \Rightarrow real$ **where**
Φ (*n,l*) = (*if n* $\geq$ *f2′∗l then ai∗(n − f2′∗l) else*
  if n $\leq$ *f1′∗l* $\wedge$ *l0* $\leq$ *l/c then ad∗(f1′∗l − n) else 0*)

lemma *Phi-Psi*: Φ (*n,l*) = Ψ (*l0* $\leq$ *l/c*) *ai ad* (*f1′∗l*) (*f2′∗l*) *n*
by(*simp*)

fun *invar* **where**
invar(n,l) = (*l* $\geq$ *l0* $\wedge$ (*l/c* $\geq$ *l0* $\longrightarrow$ *f1∗l* $\leq$ *n*) $\wedge$ *n* $\leq$ *f2∗l*)

abbreviation $U \equiv \lambda f$ -. *case f of Ins* $\Rightarrow$ *ai+1* | *Del* $\Rightarrow$ *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 (*standard, goal-cases*)
  case *1* **show** *?case* **by** (*auto simp*: *field-simps*)

next
  case (*2 s f*)
  obtain *n l* **where** [*simp*]: *s* = (*n,l*) **by** *fastforce*
  from *2* **have** *l0* ≤ *l* **and** *n* ≤ *f2∗l* **by** *auto*
  hence [*arith*]: *l* > *0* **by** *arith*
  show *?case*
  proof (*cases f*)
    case [*simp*]: *Ins*
    show *?thesis*
    proof *cases*
      assume *n+1* ≤ *f2∗l* **thus** *?thesis* **using** *2* **by** (*auto*)
    next
      assume *0*: ¬ *n+1* ≤ *f2∗l*
      have *f1*: *f1* ∗ (*e∗l*) ≤ *n+1*
      proof −
        have *f1∗e* < *f2* **using** *f1f2e* **by**(*simp add*: *field-simps*)
        hence *f1* ∗ (*e∗l*) ≤ *f2∗l* **by** *simp*
        with *0* **show** *?thesis* **by** *linarith*
      qed
      have *f2*: *n+1* ≤ *f2∗e∗l*
      proof −
        have *n+1* ≤ *f2∗l+1* **using** ‹*n* ≤ *f2∗l*› **by** *linarith*
        also have *1* = *f2∗(e−1)∗(1/(f2∗(e−1)))* **by**(*simp*)
        also note *l0f2e*
        also note ‹*l0* ≤ *l*›
        finally show *?thesis* **by** *simp* (*simp add*: *algebra-simps*)
      qed
      have *l* ≤ *l∗e* **by** *simp*
      hence *l0* ≤ *l* ∗ *e* **using** ‹*l0≤l*› **by** *linarith*
      with *0* *f1* *f2* **show** *?thesis* **by**(*simp add*: *field-simps*)
    qed
  next
    case [*simp*]: *Del*
    show *?thesis*
    proof *cases*
      assume *f1∗l* ≤ *real* (*n* − *1*)
      thus *?thesis* **using** *2* **by**(*auto*)
    next
      assume *0*: ¬ *f1∗l* ≤ *real* (*n* − *1*)
      show *?thesis*
      proof *cases*
        assume *l*: *l0* ≤ *l/c*
        hence *f1*: *f1∗(l/c)* ≤ *n−1* **using** *f1-l0*[*OF l*] *2* **by** *simp linarith*
        have *n* − *1* ≤ *f2* ∗ (*l/c*)
        proof −
          have *f1∗l* ≤ *f2∗(l/c)* **using** *f1cf2* **by** (*simp add*: *field-simps*)
          thus *?thesis* **using** *0* **by** *linarith*
        qed
        with *l* *0* *f1* **show** *?thesis* **by** (*auto*)

6

     next
       assume ¬ *l0* ≤ *l/c*
       with *2* **show** *?thesis* **by** (*auto simp add*: *field-simps*)
     qed
   qed
 qed
next
 case (*3 s*) **thus** *?case* **by**(*cases s*)(*simp split*: *if-splits*)
next
 case *4* **show** *?case* **by**(*simp add*: *field-simps not-le*)
next
 case (*5 s f*)
 obtain *n l* **where** [*simp*]: *s* = (*n,l*) **by** *fastforce*
 have [*arith*]: *l* ≥ *l0* *n* ≤ *f2*∗*l* **using** *5* **by** *auto*
 show *?case*
 proof (*cases f*)
  case [*simp*]: *Ins*
  show *?thesis* (**is** *?A* ≤ -)
  proof *cases*
   assume *n+1* ≤ *f2*∗*l*
   thus *?thesis* **by**(*simp del*: Φ.*simps* Ψ.*simps add*: *Phi-Psi Psi-diff-Ins*)
  next
   assume [*arith*]: ¬ *n+1* ≤ *f2*∗*l*
   have (*f2* − *f2′*)∗*l* ≥ *1*
    using *mult-mono*[*OF order-refl* ‹*l*≥*l0*›, *of f2*−*f2′*] *f2′-less-f2 f2f2′*
    by (*simp add*: *field-simps*)
   hence *n* ≥ *f2′*∗*l* **by**(*simp add*: *algebra-simps*)
   hence *Phi*: Φ *s* = *ai* ∗ (*n* − *f2′*∗*l*) **by** *simp*
   have *f1′*∗*e*∗*l* ≤ *f2*∗*l* **using** *f1′ef2* **by**(*simp*)
   hence *f1′*∗*e*∗*l* < *n+1* **by** *linarith*
   have *?A* ≤ *n* − *ai*∗(*f2* − *f2′*)∗*l* + *ai* + *1*
   proof *cases*
    assume *n+1* < *f2′*∗(*e*∗*l*)
    hence *?A* = *n+1* − *ai*∗(*n* − *f2′*∗*l*) **using** *Phi* ‹*f1′*∗*e*∗*l* < *n+1*› **by** *simp*
    also have ... = *n* + *ai*∗(−(*n+1*) + *f2′*∗*l*) + *ai+1*
     by(*simp add*: *algebra-simps*)
    also have −(*n+1*) ≤ −*f2*∗*l* **by** *linarith*
    finally show *?thesis* **by**(*simp add*: *algebra-simps*)
   next
    assume ¬ *n+1* < *f2′*∗(*e*∗*l*)
    hence *?A* = *n* + *ai*∗(−*f2′*∗*e* + *f2′*)∗*l* + *ai+1* **using** *Phi*
     by(*simp add*: *algebra-simps*)
    also have −*f2′*∗*e* ≤ −*f2* **using** *f2-le-f2′e* **by** *linarith*
    finally show *?thesis* **by**(*simp add*: *algebra-simps*)
   qed
   also have ... = *n* − *f2*∗*l* + *ai+1* **using** *f2′-less-f2* **by**(*simp add*: *ai-def*)
   finally show *?thesis* **by** *simp*
  qed
 next

7

case [*simp*]: *Del*

show *?thesis* (**is** *?A ≤ -*)

proof *cases*

  assume *n=0* **with** *5* **show** *?thesis*

    by(*simp add: mult-le-0-iff field-simps*)

next

  assume [*arith*]: *n≠0*

  show *?thesis*

  proof *cases*

    assume *real n − 1 ≥ f1∗l ∨ l/c < l0*

    thus *?thesis* **using** *f1′-le-f2′*

      by(*auto simp del:* Φ.*simps* Ψ.*simps simp add: Phi-Psi Psi-diff-Del*)

  next

    assume ¬ (*real n − 1 ≥ f1∗l ∨ l/c < l0*)

    hence [*arith*]: *real n − 1 < f1∗l l/c ≥ l0* **by** *linarith+*

    hence *l ≥ l0∗c* **and** *l/c ≥ l0* **and** *f1∗l ≤ n* **using** *5*

      by (*auto simp: field-simps*)

    have *f1∗l ≤ f2′∗l/c* **using** *f1f2′c* **by**(*simp add: field-simps*)

    hence *f2*: *n−1 < f2′∗l/c* **by** *linarith*

    have *f1′∗l ≤ f2′∗l* **using** *f1′-le-f2′* **by** *simp*

    have *(f1′ − f1)∗l ≥ 1*

      using *mult-mono*[*OF order-refl* ‹*l≥l0∗c*›, *of f1′−f1*] *f1-less-f1′ f1′f1*

      by (*simp add: field-simps*)

    hence *n < f1′∗l* **by**(*simp add: algebra-simps*)

    hence *Phi*: Φ *s = ad∗(f1′∗l − n)*

      apply(*simp*) **using** ‹*f1′∗l ≤ f2′∗l*› **by** *linarith*

    have *?A ≤ n − ad∗(f1′ − f1)∗l + ad*

    proof *cases*

      assume *n−1 < f1′∗l/c ∧ l/(c∗c) ≥ l0*

      hence Φ *(nxt f s) = ad∗(f1′∗l/c − (n−1))* **using** *f2* **by**(*auto*)

      hence *?A = n + ad∗(f1′∗l/c − (n−1)) − (ad∗(f1′∗l − n))*

        using *Phi* **by** (*simp add: algebra-simps*)

      also have *. . . = n + ad∗(f1′/c − f1′)∗l + ad*

        by(*simp add: algebra-simps*)

      also note *f1′c-le-f1*

      finally show *?thesis* **by**(*simp add: algebra-simps*)

    next

      assume ¬(*n−1 < f1′∗l/c ∧ l/(c∗c) ≥ l0*)

      hence Φ *(nxt f s) = 0* **using** *f2* **by**(*auto*)

      hence *?A = n + ad∗(n − f1′∗l)* **using** *Phi*

        by (*simp add: algebra-simps*)

      also have *. . . = n + ad∗(n−1 − f1′∗l) + ad* **by**(*simp add: algebra-simps*)

      also have *n−1 ≤ f1∗l* **by** *linarith*

      finally show *?thesis* **by** (*simp add: algebra-simps*)

    qed

    also have *. . . = n − f1∗l + ad* **using** *f1-less-f1′* **by**(*simp add: ad-def*)

    finally show *?thesis* **by** *simp*

  qed

qed

8

   qed
qed

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*: $(e*c)/(f2*(min\ e\ c\ -\ 1)) \le l0$
begin

lemma *l0e*: $(e*c)/(f2*(e-1)) \le l0$
proof −
  have *0*: $f2 * (l0 * min\ e\ c) \le e * (f2 * l0)$
    by (*simp add*: *min-def ac-simps mult-right-mono*)
  from *l0* **show** *?thesis* **apply**(*simp add*: *field-simps*) **using** *0* **by** *linarith*
qed

lemma *l0c*: $(e*c)/(f2*(c-1)) \le l0$
proof −
  have *0*: $f2 * (l0 * min\ e\ c) \le c * (f2 * l0)$
    by (*simp add*: *min-def ac-simps mult-right-mono*)
  from *l0* **show** *?thesis* **apply**(*simp add*: *field-simps*) **using** *0* **by** *linarith*
qed

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 (*standard, goal-cases*)
  case *1* **show** *?case* **by**(*rule e1*)
next
  case *2* **show** *?case* **by**(*rule c1*)
next
  case *3* **show** *?case* **by**(*simp*)
next
  case *4* **show** *?case* **by**(*simp add*: *field-simps*)
next
  case *5* **show** *?case* **by**(*simp add*: *field-simps*)
next
  case *6* **show** *?case* **by**(*simp*)
next
  case *7* **show** *?case* **by**(*simp*)
next
  case *8* **show** *?case* **using** *l0e less-1-mult*[*OF c1 e1*] **by**(*simp add*: *field-simps*)
next
  case *9* **show** *?case* **using** *l0c* **by**(*simp*)

9

next
  case *10* **show** *?case*
  proof−
    have *1*: *c∗e>e* **by** (*simp*)
    show *?thesis* **using** *l0e* **apply**(*simp add*: *field-simps*) **using** *1* **by** *linarith*
  qed
next
  case *11* **show** *?case*
  proof−
    have *1*: *c∗e>e* **by** (*simp*)
    show *?thesis* **using** *l0c*
      apply(*simp add*: *algebra-simps pos-le-divide-eq*)
      apply(*simp add*: *field-simps*)
      using *1* **by** *linarith*
  qed
qed

lemma *ai = e/(e−1)*
unfolding *ai-def* **by**(*simp add*: *field-simps*)

lemma *ad = 1/(c−1)*
unfolding *ad-def* **by**(*simp add*: *field-simps*)

end

interpretation *I1*: *Optimal* **where** *e=2* **and** *c=2* **and** *f2=1* **and** *l0=4*
proof qed *simp-all*

interpretation *I2*: *Optimal* **where** *e=2* **and** *c=2* **and** *f2=3/4* **and** *l0=6*
proof qed *simp-all*

interpretation *I3*: *Optimal* **where** *e=2* **and** *c=2* **and** *f2=0.8* **and** *l0=5*
proof qed *simp-all*

interpretation *I4*: *Optimal* **where** *e=3* **and** *c=3* **and** *f2=0.9* **and** *l0=5*
proof qed *simp-all*

interpretation *I5*: *Optimal* **where** *e=4* **and** *c=4* **and** *f2=1* **and** *l0=6*
proof qed *simp-all*

interpretation *I6*: *Optimal* **where** *e=2.5* **and** *c=2.5* **and** *f2=1* **and** *l0=5*
proof qed *simp-all*

interpretation *I7*: *Optimal* **where** *f2=1* **and** *c=3/2* **and** *e=2* **and** *l0=6*
proof qed (*simp-all add*: *min-def*)

interpretation *I8*: *Optimal* **where** *f2=1* **and** *e=3/2* **and** *c=2* **and** *l0=6*
proof qed (*simp-all add*: *min-def*)

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 −
  have $0 < 1/(f2{*}(e{-}1))$ **by**(*simp*)
  also note *l0f2e*
  finally show *?thesis* **by** *simp*
qed

lemma *f1-l0*: **assumes** $l0 \leq l/c$ **shows** $f1{*}(l/c) \leq f1{*}l - 1$
proof −
  have $1 = f1{*}((c{-}1)/c){*}(c{*}(1/(f1{*}(c{-}1))))$
    using *f1′-le-f2′ f2′-less-f2* **by**(*simp add: field-simps*)
  also note *l0f1c*
  also have l': $c{*}l0 \leq l$ **using** *assms*(*1*) **by**(*simp add: field-simps*)
  finally show *?thesis* **by**(*simp add: divide-le-cancel*) (*simp add: field-simps*)
qed

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$(n,l) = (l \geq l0 \land (\lfloor l/c\rfloor \geq l0 \longrightarrow f1{*}l \leq n) \land n \leq f2{*}l)$

lemma *invar-init*: *invar* $(0,l0)$
by (*auto simp*: *le-floor-iff field-simps*)

lemma *invar-pres*: **assumes** *invar s* **shows** *invar*(*nxt f s*)
proof −
  obtain *n l* **where** [*simp*]: *s = (n,l)* **by** *fastforce*
  from *assms* **have** $l0 \leq l$ **and** $n \leq f2*l$ **by** *auto*
  show *?thesis*
  proof (*cases f*)
    case [*simp*]: *Ins*
    show *?thesis*
    proof *cases*
      assume $n+1 \leq f2*l$ **thus** *?thesis* **using** *assms* **by** (*auto*)
    next
      assume *0*: ¬ $n+1 \leq f2*l$
      have *f1*: $f1 * \lceil e*l \rceil \leq n+1$
      proof −
        have $\lceil e*l \rceil \leq e*l + 1$ **by** *linarith*
        hence $f1 * \lceil e*l \rceil \leq f1 * (e*l + 1)$ **by** *simp*
        also have $\ldots \leq f2*l$
        proof −
          have $f1 \leq (f2 − f1*e)*l0$
            using *l0f2f1e f1f2e* **by**(*simp add: field-simps*)
          also note ‹$l0 \leq l$›
          finally show *?thesis* **using** *f1f2e*[*simplified field-simps*]
            by (*simp add:ac-simps mult-left-mono*) (*simp add:algebra-simps*)
        qed
        finally show *?thesis* **using** *0* **by** *linarith*
      qed
      have $n+1 \leq f2*e*l$
      proof −
        have $n+1 \leq f2*l+1$ **using** ‹$n \leq f2*l$› **by** *linarith*
        also have $1 = f2*(e−1)*(1/(f2*(e−1)))$ **by**(*simp*)
        also note *l0f2e*
        also note ‹$l0 \leq l$›
        finally show *?thesis* **by** *simp* (*simp add: algebra-simps*)
      qed
      also have $f2*e*l \leq f2*\lceil e*l \rceil$ **by** *simp*
      finally have *f2*: $n+1 \leq f2*\lceil e*l \rceil$.
      have $l < e*l$ **using** ‹$l0 \leq l$› **by** *simp*
      hence $l0 \leq e*l$ **using** ‹$l0{\leq}l$› **by** *linarith*
      with *0 f1 f2* **show** *?thesis* **by** (*auto simp add: field-simps*) *linarith*
    qed
  next
    case [*simp*]: *Del*
    show *?thesis*
    proof *cases*
      assume $f1*l \leq real\ n − 1$
      thus *?thesis* **using** *assms* **by**(*auto*)
    next
      assume *0*: ¬ $f1*l \leq real\ n − 1$

12

    show *?thesis*
    proof *cases*
      assume *n=0* **thus** *?thesis* **using** *0 assms* **by**(*simp add: field-simps*)
    next
      assume $n \neq 0$
      show *?thesis*
      proof *cases*
        assume *l*: $l0 \leq \lfloor l/c \rfloor$
        hence *l'*: $l0 \leq l/c$ **by** *linarith*
        have $f1 * \lfloor l/c \rfloor \leq f1*(l/c)$ **by**(*simp del: times-divide-eq-right*)
        hence *f1*: $f1*\lfloor l/c \rfloor \leq n-1$ **using** *l' f1-l0*[*OF l'*] *assms* $\langle n \neq 0 \rangle$
          by(*simp add: le-floor-iff*)
        have $n-1 \leq f2 * \lfloor l/c \rfloor$
        proof −
          have $n-1 < f1*l$ **using** *0* $\langle n \neq 0 \rangle$ **by** *linarith*
          also have $f1*l \leq f2*(l/c) - f2$
          proof −
            have $(f2 - f1*c)*l0 \geq f2$
              using *l0f2f1c f1cf2* **by**(*simp add: field-simps*)
            with *mult-left-mono*[*OF* $\langle l0 \leq l/c \rangle$, *of f2−f1*c*] *f1cf2*
            have $(f2 - f1*c)*(l/c) \geq f2$ **by** *linarith*
            thus *?thesis* **by**(*simp add: field-simps*)
          qed
          also have $\ldots \leq f2*\lfloor l/c \rfloor$
          proof −
            have $l/c - 1 \leq \lfloor l/c \rfloor$ **by** *linarith*
            from *mult-left-mono*[*OF this, of f2*] **show** *?thesis*
              by(*simp add: algebra-simps*)
          qed
          finally show *?thesis* **using** *0* $\langle n \neq 0 \rangle$ **by** *linarith*
        qed
        with *l 0 f1* $\langle n \neq 0 \rangle$ **show** *?thesis* **by** (*auto*)
      next
        assume $\neg \; l0 \leq \lfloor l/c \rfloor$
        with *0 assms* **show** *?thesis* **by** (*auto simp add: field-simps*)
      qed
    qed
  qed
 qed
qed

end

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

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

13

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

lemma *aigr0*[*arith*]: $ai > 1$
using *f2'-less-f2* **by**(*simp add*: *ai-def field-simps*)

lemma *adgr0*[*arith*]: $ad > 0$
using *f1-less-f1'* **by**(*simp add*: *ad-def field-simps*)

lemma *f1'ad*[*arith*]: $f1'*ad > 0$
by *simp*

lemma *f2'ai*[*arith*]: $f2'*ai > 0$
by *simp*

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

lemma *Phi-Psi*: Φ (*n,l*) = Ψ ($l0 \leq \lfloor l/c \rfloor$) *ai ad* ($f1'*l$) ($f2'*l$) *n*
by(*simp*)

abbreviation $U \equiv \lambda f$ -. *case f of Ins* $\Rightarrow$ $ai+1 + f1'*ad$ | *Del* $\Rightarrow$ $ad+1 + f2'*ai$

interpretation *tb*: *Amortized*
 where *init* = (0,*l0*) **and** *nxt* = *nxt*
 and *inv* = *invar*
 and $T = T$ **and** $\Phi = \Phi$
 and $U = U$
proof (*standard, goal-cases*)
 case *1* **show** *?case* **by** (*fact invar-init*)
next
 case *2* **thus** *?case* **by**(*fact invar-pres*)
next
 case (*3 s*) **thus** *?case* **by**(*cases s*)(*simp split*: *if-splits*)
next
 case *4* **show** *?case*
  by(*auto simp*: *field-simps mult-le-0-iff le-floor-iff*)
next
 case (*5 s f*)
 obtain *n l* **where** [*simp*]: $s = (n,l)$ **by** *fastforce*
 show *?case*
 proof (*cases f*)
  case [*simp*]: *Ins*
  show *?thesis* (**is** *?A* $\leq$ -)
  proof *cases*
   assume $n+1 \leq f2*l$
   hence *?A* $\leq$ $ai+1$ **by**(*simp del*: Φ.*simps* Ψ.*simps add*: *Phi-Psi Psi-diff-Ins*)
   thus *?thesis* **by** *simp*
  next

assume [*arith*]: ¬ *n+1* ≤ *f2∗l*
have [*arith*]: *l* ≥ *l0*   *n* ≤ *f2∗l* **using** *5* **by** *auto*
have (*f2* − *f2′*)∗*l* ≥ *1*
  using *mult-mono*[*OF order-refl, of l0 l f2−f2′*] *f2′-less-f2 f2f2′*
  by (*simp add: field-simps*)
hence *n* ≥ *f2′∗l* **by**(*simp add: algebra-simps*)
hence *Phi*: Φ *s* = *ai* ∗ (*n* − *f2′∗l*) **by** *simp*
have [*simp*]: *real* (*nat* ⌈*e∗l*⌉) = *real-of-int* ⌈*e∗l*⌉
  by (*simp add: order.order-iff-strict*)
have *?A* ≤ *n* − *ai∗*(*f2* − *f2′*)∗*l* + *ai* + *1* + *f1′∗ad* (**is** - ≤ *?R*)
proof *cases*
  assume *f2′*: *n+1* < *f2′∗*⌈*e∗l*⌉
  show *?thesis*
  proof *cases*
    assume *n+1* ≤ *f1′∗*⌈*e∗l*⌉
    hence *?A* ≤ *n+1* + *ad∗*(*f1′∗*⌈*e∗l*⌉−(*n+1*)) − *ai∗*(*n* − *f2′∗l*)
      using *Phi f2′* **by** *simp*
    also have *f1′∗*⌈*e∗l*⌉ − (*n+1*) ≤ *f1′*
    proof −
      have *f1′∗*⌈*e∗l*⌉ ≤ *f1′∗*(*e∗l* + *1*) **by**(*simp*)
      also have . . . = *f1′∗e∗l* + *f1′* **by**(*simp add: algebra-simps*)
      also have *f1′∗e∗l* ≤ *f2∗l* **using** *f1′ef2* **by**(*simp*)
      finally show *?thesis* **by** *linarith*
    qed
 also have *n+1+ad∗f1′−ai∗*(*n−f2′∗l*) = *n+ai∗*(−*real*(*n+1*)+*f2′∗l*)+*ai+f1′∗ad+1*
    by(*simp add: algebra-simps*)
    also have −*real*(*n+1*) ≤ −*f2∗l* **by** *linarith*
    finally show *?thesis* **by**(*simp add: algebra-simps*)
  next
    assume ¬ *n+1* ≤ *f1′∗*⌈*e∗l*⌉
    hence *?A* = *n+1* − *ai∗*(*n* − *f2′∗l*) **using** *Phi f2′* **by** (*simp*)
    also have *n+1−ai∗*(*n−f2′∗l*) = *n+ai∗*(−*real*(*n+1*)+*f2′∗l*)+*ai+1*
      by(*simp add: algebra-simps*)
    also have −*real*(*n+1*) ≤ −*f2∗l* **by** *linarith*
    also have *n+ai∗*(−*f2∗l+f2′∗l*)+*ai+1* ≤ *?R*
      by(*simp add: algebra-simps*)
    finally show *?thesis* **by**(*simp*)
  qed
 next
  assume ¬ *n+1* < *f2′∗*⌈*e∗l*⌉
  hence *?A* = *n* + *ai∗*(−*f2′∗*⌈*e∗l*⌉ + *f2′∗l*) + *ai+1* **using** *Phi*
    by(*simp add: algebra-simps*)
  also have −*f2′∗*⌈*e∗l*⌉ ≤ −*f2′∗e∗l* **by**(*simp*)
  also have −*f2′∗e* ≤ −*f2* **using** *f2-le-f2′e* **by** *linarith*
  also have *n+ai∗*(−*f2∗l+f2′∗l*)+*ai+1* ≤ *?R* **by**(*simp add: algebra-simps*)
  finally show *?thesis* **by**(*simp*)
qed
also have . . . = *n* − *f2∗l* + *ai+f1′∗ad+1* **using** *f2′-less-f2*
  by(*simp add: ai-def*)

    finally show *?thesis* **by** *simp*
  qed
next
  case [*simp*]: *Del*
  have [*arith*]: *l* ≥ *l0* **using** *5* **by** *simp*
  show *?thesis*
  proof *cases*
    assume *n=0* **with** *5* **show** *?thesis*
      by(*simp add*: *mult-le-0-iff field-simps*)
  next
    assume [*arith*]: *n≠0*
    show *?thesis* (**is** *?A* ≤ -)
    proof *cases*
      assume *real n − 1* ≥ *f1∗l* ∨ ⌊*l/c*⌋ < *l0*
      hence *?A* ≤ *ad+1* **using** *f1′-le-f2′*
        by(*auto simp del*: Φ.*simps* Ψ.*simps simp add*: *Phi-Psi Psi-diff-Del*)
      thus *?thesis* **by** *simp*
    next
      assume ¬ (*real n − 1* ≥ *f1∗l* ∨ ⌊*l/c*⌋ < *l0*)
      hence *n*: *real n − 1* < *f1∗l* **and** *lc′*: ⌊*l/c*⌋ ≥ *l0* **and** *lc*: *l/c* ≥ *l0*
        by *linarith+*
      have *f1′∗l* ≤ *f2′∗l* **using** *f1′-le-f2′* **by** *simp*
      have (*f1′ − f1*)∗*l* ≥ *1* **using** *mult-mono*[*OF order-refl, of l0 l/c f1′−f1*]
        lc f1-less-f1′ f1′f1 **by** (*simp add*: *field-simps*)
      hence *n* < *f1′∗l* **using** *n* **by**(*simp add*: *algebra-simps*)
      hence *Phi*: Φ *s* = *ad∗(f1′∗l − n)*
        apply(*simp*) **using** ‹*f1′∗l* ≤ *f2′∗l*› *lc* **by** *linarith*
      have *?A* ≤ *n − ad∗(f1′ − f1)∗l + ad + f2′∗ai* (**is** *- ≤ ?R + -*)
      proof *cases*
        assume *f2′*: *n−1* < *f2′∗*⌊*l/c*⌋
        show *?thesis*
        proof *cases*
          assume *n−1* < *f1′∗*⌊*l/c*⌋ ∧ ⌊⌊*l/c*⌋*/c*⌋ ≥ *l0*
          hence Φ (*nxt f s*) = *ad∗(f1′∗*⌊*l/c*⌋ − (*n−1*)) **using** *f2′ n lc′* **by**(*auto*)
          hence *?A* = *n + ad∗(f1′∗*⌊*l/c*⌋ − (*n−1*)) − (*ad∗(f1′∗l − n*))
            using *Phi n lc′* **by** (*simp add*: *algebra-simps*)
          also have ⌊*l/c*⌋ ≤ *l/c* **by**(*simp*)
        also have *n+ad∗(f1′∗(l/c)−(n−1))−(ad∗(f1′∗l−n))* = *n+ad∗(f1′/c−f1′)∗l+ad*
          by(*simp add*: *algebra-simps*)
          also note *f1′c-le-f1*
          finally have *?A* ≤ *?R* **by**(*simp add*: *algebra-simps*)
          thus *?thesis* **by** *linarith*
        next
          assume ¬(*n−1* < *f1′∗*⌊*l/c*⌋ ∧ ⌊⌊*l/c*⌋*/c*⌋ ≥ *l0*)
          hence Φ (*nxt f s*) = *0* **using** *f2′ n lc′* **by**(*auto*)
          hence *?A* = *n + ad∗(n − f1′∗l)* **using** *Phi n lc′*
            by (*simp add*: *algebra-simps*)
        also have ... = *n + ad∗(n−1 − f1′∗l) + ad* **by**(*simp add*: *algebra-simps*)
          also have *n−1* ≤ *f1∗l* **using** *n* **by** *linarith*

finally have *?A ≤ ?R* **by** (*simp add*: *algebra-simps*)
        thus *?thesis* **by** *linarith*
      qed
    next
      assume *f2′*: ¬ *n−1 < f2′∗⌊l/c⌋*
      hence *?A = n + ai∗(n−1−f2′∗⌊l/c⌋) − ad∗(f1′∗l − n)*
        using *Phi n lc′* **by** (*simp*)
      also have *n−1−f2′∗⌊l/c⌋ ≤ f2′*
      proof −
        have *f1∗l ≤ f2′∗(l/c)* **using** *f1f2′c* **by**(*simp add*: *field-simps*)
        hence *n−1 < f2′∗(l/c)* **using** *n* **by** *linarith*
        also have *l/c ≤ ⌊l/c⌋ + 1* **by** *linarith*
        finally show *?thesis* **by**(*fastforce simp*: *algebra-simps*)
      qed
        also have *n+ai∗f2′−ad∗(f1′∗l−n) = n + ad∗(n−1 − f1′∗l) + ad + f2′∗ai*

f2′∗ai

        by(*simp add*: *algebra-simps*)
      also have *n−1 ≤ f1∗l* **using** *n* **by** *linarith*
      finally show *?thesis* **by**(*simp add*: *algebra-simps*)
    qed
      also have *. . . = n − f1∗l + ad + f2′∗ai* **using** *f1-less-f1′* **by**(*simp add*: *ad-def*)
    finally show *?thesis* **using** *n* **by** *simp*
  qed
  qed
  qed
qed

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*

using *f1-less-f1″ f1* **by** *linarith*

lemma *f2″-gr0*[*arith*]: *f2″ > 0*
using *f2′ f2′-less-f2″* **by** *linarith*

lemma *aigr0*[*arith*]: *ai > 0*
using *f2″-less-f2* **by**(*simp add*: *ai-def field-simps*)

lemma *adgr0*[*arith*]: *ad > 0*
using *f1-less-f1″* **by**(*simp add*: *ad-def field-simps*)

fun Φ :: *nat ∗ nat ⇒ real* **where**
$\Phi(n,l) = ($ *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*
by(*simp*)

abbreviation *U ≡ λf -. 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 (*standard, goal-cases*)
  case *1* **show** *?case* **by** (*fact invar-init*)
next
  case *2* **thus** *?case* **by**(*fact invar-pres*)
next
  case (*3 s*) **thus** *?case* **by**(*cases s*)(*simp split*: *if-splits*)
next
  case *4* **show** *?case*
    by(*auto simp*: *field-simps mult-le-0-iff le-floor-iff*)
next
  case (*5 s f*)
  obtain *n l* **where** [*simp*]: *s = (n,l)* **by** *fastforce*
  show *?case*
  proof (*cases f*)
    case [*simp*]: *Ins*
    show *?thesis* (**is** *?L ≤ -*)
    proof *cases*
      assume *n+1 ≤ f2∗l*
      thus *?thesis* **by**(*simp del*: Φ*.simps* Ψ*.simps add*: *Phi-Psi Psi-diff-Ins*)
    next
      assume [*arith*]: ¬ *n+1 ≤ f2∗l*
      have [*arith*]: *l ≥ l0 n ≤ f2∗l* **using** *5* **by** *auto*
      have *l0 ≤ e∗l* **using** ⟨*l0 ≤ l*⟩ *e1 mult-mono*[*of 1 e l0 l*] **by** *simp*
      have *(f2 − f2″)∗l ≥ 1*

using *mult-mono[OF order-refl, of l0 l f2−f2″] f2″-less-f2 f2f2″*

  by (*simp add*: *algebra-simps*)

hence $n \geq f2″*l$ **by**(*simp add*: *algebra-simps*)

hence *Phi*: $\Phi \; s = ai * (n - f2″*l)$ **by** *simp*

have [*simp*]: *real* (*nat* $\lceil e*l \rceil$) = *real-of-int* $\lceil e*l \rceil$

  by (*simp add*: *order.order-iff-strict*)

have $?L \leq n - ai*(f2 - f2″)*l + ai + 1$ (**is** - $\leq$ *?R*)

proof *cases*

  assume *f2″*: $n+1 < f2″*\lceil e*l \rceil$

  have $f1″*\lceil e*l \rceil \leq f1″*(e*l + 1)$ **by**(*simp*)

  also note *f1″-f1′[OF* ‹$l0 \leq e*l$›*]*

  also have $f1′*(e*l) \leq f2*l$ **using** *f1′ef2* **by**(*simp*)

  also have $f2*l \leq n+1$ **by** *linarith*

  finally have $?L \leq n+1 - ai*(n - f2″*l)$

    using *Phi f2″* **by** (*simp*)

  also have $n+1-ai*(n-f2″*l) = n+ai*(-real(n+1)+f2″*l)+ai+1$

    by(*simp add*: *algebra-simps*)

  also have $-real(n+1) \leq -f2*l$ **by** *linarith*

  finally show *?thesis* **by**(*simp add*: *algebra-simps*)

  next

  assume $\neg \; n+1 < f2″*\lceil e*l \rceil$

  hence $?L = n + ai*(-f2″*\lceil e*l \rceil + f2″*l) + ai+1$ **using** *Phi*

    by(*simp add*: *algebra-simps*)

  also have $-f2″*\lceil e*l \rceil \leq -f2″*e*l$ **by**(*simp*)

  also have $-f2″*e \leq -f2′*e$ **using** *f2′-less-f2″* **by**(*simp*)

  also have $-f2′*e \leq -f2$ **using** *f2-le-f2′e* **by**(*simp*)

  also have $n+ai*(-f2*l+f2″*l)+ai+1 \leq \; ?R$ **by**(*simp add*: *algebra-simps*)

  finally show *?thesis* **by**(*simp*)

  qed

  also have $\ldots = n - f2*l + ai+1$ **using** *f2″-less-f2*

    by(*simp add*: *ai-def*)

  finally show *?thesis* **by** *simp*

qed

next

case [*simp*]: *Del*

have [*arith*]: $l \geq l0$ **using** *5* **by** *simp*

show *?thesis*

proof *cases*

  assume $n=0$ **with** *5* **show** *?thesis*

    by(*simp add*: *mult-le-0-iff field-simps*)

  next

  assume [*arith*]: $n \neq 0$

  show *?thesis* (**is** $?A \leq$ -)

  proof *cases*

    assume *real* $n - 1 \geq f1*l \vee \lfloor l/c \rfloor < l0$

    thus *?thesis* **using** *f1″-less-f1′ f1′-le-f2′ f2′-less-f2″*

      by(*auto simp del*: Φ*.simps* Ψ*.simps simp add*: *Phi-Psi Psi-diff-Del*)

    next

    assume $\neg$ (*real* $n - 1 \geq f1*l \vee \lfloor l/c \rfloor < l0$)

hence *n*: *real n* − *1* < *f1*∗*l* **and** *lc'*: ⌊*l*/*c*⌋ ≥ *l0* **and** *lc*: *l*/*c* ≥ *l0*
  by *linarith+*
have *f1''*∗*l* ≤ *f2''*∗*l*
  using *f1''-less-f1' f1'-le-f2' f2'-less-f2''* **by** *simp*
have (*f1''* − *f1*)∗*l* ≥ *1*
  using *mult-mono[OF order-refl, of l0 l/c f1''−f1] lc f1-less-f1'' f1''f1*
  by (*simp add*: *field-simps*)
hence *n* < *f1''*∗*l* **using** *n* **by**(*simp add*: *algebra-simps*)
hence *Phi*: Φ *s* = *ad*∗(*f1''*∗*l* − *n*)
  apply(*simp*) **using** ⟨*f1''*∗*l* ≤ *f2''*∗*l*⟩ *lc* **by** *linarith*
have *f2'*: *n−1* < *f2''*∗⌊*l*/*c*⌋
proof −
  have *n−1* < *f1*∗*l* **using** *n* **by** *linarith*
  also have *f1*∗*l* ≤ *f2'*∗(*l*/*c*) **using** *f1f2'c* **by**(*auto simp*: *field-simps*)
  also note *f2'-f2''[OF ⟨l/c≥l0⟩]*
  also have *f2''*∗(*l*/*c* − *1*) ≤ *f2''*∗⌊*l*/*c*⌋ **by** *simp*
  finally show *?thesis* **by**(*simp*)
qed
have *?A* ≤ *n* − *ad*∗(*f1''* − *f1*)∗*l* + *ad*
proof *cases*
  assume *n−1* < *f1''*∗⌊*l*/*c*⌋ ∧ ⌊⌊*l*/*c*⌋/*c*⌋ ≥ *l0*
  hence Φ (*nxt f s*) = *ad*∗(*f1''*∗⌊*l*/*c*⌋ − (*n−1*)) **using** *f2' n lc'* **by**(*auto*)
  hence *?A* = *n* + *ad*∗(*f1''*∗⌊*l*/*c*⌋ − (*n−1*)) − (*ad*∗(*f1''*∗*l* − *n*))
    using *Phi n lc'* **by** (*simp add*: *algebra-simps*)
  also have ⌊*l*/*c*⌋ ≤ *l*/*c* **by**(*simp*)
 also have *n*+*ad*∗(*f1''*∗(*l*/*c*)−(*n−1*))−(*ad*∗(*f1''*∗*l*−*n*)) = *n*+*ad*∗(*f1''*/*c*−*f1''*)∗*l*+*ad*
    by(*simp add*: *algebra-simps*)
  also have *f1''*/*c* ≤ *f1'*/*c* **using** *f1''-less-f1'* **by**(*simp add*: *field-simps*)
  also note *f1'c-le-f1*
  finally show *?thesis* **by**(*simp add*: *algebra-simps*)
next
  assume ¬(*n−1* < *f1''*∗⌊*l*/*c*⌋ ∧ ⌊⌊*l*/*c*⌋/*c*⌋ ≥ *l0*)
  hence Φ (*nxt f s*) = *0* **using** *f2' n lc'* **by**(*auto*)
  hence *?A* = *n* + *ad*∗(*n* − *f1''*∗*l*) **using** *Phi n lc'*
    by (*simp add*: *algebra-simps*)
  also have . . . = *n* + *ad*∗(*n−1* − *f1''*∗*l*) + *ad* **by**(*simp add*: *algebra-simps*)
  also have *n−1* ≤ *f1*∗*l* **using** *n* **by** *linarith*
  finally show *?thesis* **by** (*simp add*: *algebra-simps*)
qed
also have . . . = *n* − *f1*∗*l* + *ad* **using** *f1-less-f1''* **by**(*simp add*: *ad-def*)
finally show *?thesis* **using** *n* **by** *simp*
  qed
  qed
 qed
qed

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* ≥ (*f2*+*1*)/(*f2*−*f2′*)
assumes *l0-f1f1′*: *l0* ≥ (*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 −
  have *f2*/(*f2*−*f2′*) ≥ *1* **using** *f2′-less-f2* **by**(*simp add*: *field-simps*)
  thus *?thesis* **using** *l0-f2-f2′ f2′-less-f2* **by** *linarith*
qed

lemma *f1′′-less-f1′*: *f1′′* < *f1′*
by(*simp add*: *f1′′-def field-simps*)

lemma *f1-less-f1′′*: *f1* < *f1′′*
proof −
  have *1* + *l0* > *0* **by** (*simp add*: *add-pos-pos*)
  hence *f1′′*> *f1* ⟷ *l0* > *f1*/((*f1′*−*f1*))
    using *f1-less-f1′* **by**(*simp add*: *f1′′-def field-simps*)
  also have . . . ⟷ *True* **using** *l0-f1-f1′* **by** *blast*
  finally show *?thesis* **by** *blast*
qed

lemma *f2′-less-f2′′*: *f2′* < *f2′′*
using *l0-gr1* **by**(*simp add*: *f2′′-def field-simps*)

lemma *f2′′-less-f2*: *f2′′* < *f2*
proof −
  have *f2′′*< *f2* ⟷ *l0* > *f2*/(*f2*−*f2′*)
    using *f2′-less-f2 l0-gr1* **by**(*simp add*: *f2′′-def field-simps*)
  also have . . . ⟷ *True* **using** *l0-f2-f2′* **by** *blast*
  finally show *?thesis* **by** *blast*
qed

lemma *f2f2′′*: (*f2* − *f2′′*)∗*l0* ≥ *1*
proof −
  have (*f2* − *f2′′*)∗(*l0*−*1*) ≥ *1*
    using *l0-gr1 l0-f2f2′ f2′-less-f2*
    by(*simp add*: *f2′′-def algebra-simps del*: *of-nat-diff*) (*simp add*: *field-simps*)
  thus *?thesis* **using** *f2′′-less-f2* **by** (*simp add*: *algebra-simps*)

qed

lemma *f1″f1*: $(f1″ − f1){*}c{*}l0 \geq 1$
proof −
  have $1 \leq (f1′ − f1){*}c{*}l0 − f1′{*}c$ **using** *l0-f1f1′ f1-less-f1′*
    by(*simp add*: *field-simps*)
  also have $\ldots = (f1′{*}((l0−1)/l0) − f1){*}c{*}l0$
    by(*simp add*: *field-simps*)
  also have $(l0−1)/l0 \leq l0/(l0+1)$
    by(*simp add*: *field-simps*)
  also have $f1′{*}(l0/(l0+1)) = f1′{*}l0/(l0+1)$
    by(*simp add*: *algebra-simps*)
  also note *f1″-def*[*symmetric*]
  finally show *?thesis* **by**(*simp*)
qed

lemma *f1″-f1′*: **assumes** $l \geq real\ l0$ **shows** $f1″{*}(l+1) \leq f1′ * l$
proof −
  have $f1″{*}(l+1) = f1′{*}(l0/(l0+1)){*}(l+1)$
    by(*simp add*: *f1″-def field-simps*)
  also have $l0/(l0+1) \leq l/(l+1)$ **using** *assms*
    by(*simp add*: *field-simps*)
  finally show *?thesis* **using** ‹$l0 \leq l$› **by**(*simp*)
qed

lemma *f2′-f2″*: **assumes** $l \geq real\ l0$ **shows** $f2′ * l \leq f2″ * (l−1)$
proof −
  have $f2′ * l = f2′ * l + f2′{*}((l0−1)/(l0−1) − 1)$ **using** *l0-gr1* **by** *simp*
  also have $(l0−1)/(l0−1) \leq (l−1)/(l0−1)$ **using** ‹$l{\geq}l0$› **by**(*simp*)
  also have $f2′{*}l + f2′{*}((l−1)/(l0−1) − 1) = f2″{*}(l−1)$
    using *l0-gr1* **by**(*simp add*: *f2″-def field-simps*)
  finally show *?thesis* **by** *simp*
qed

sublocale *Table2*
proof
qed (*fact f1-less-f1″ f1″-less-f1′ f2′-less-f2″ f2″-less-f2 f1″f1 f2f2″ f1″-f1′ f2′-f2″*)+

end

end

References

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