The Akra–Bazzi theorem and the Master theorem

Manuel Eberl

June 17, 2024

Abstract

This article contains a formalisation of the Akra–Bazzi method [1] based on a proof by Leighton [2]. It is a generalisation of the well-known Master Theorem for analysing the complexity of Divide & Conquer algorithms. We also include a generalised version of the Master theorem based on the Akra–Bazzi theorem, which is easier to apply than the Akra–Bazzi theorem itself.

Some proof methods that facilitate applying the Master theorem are also included. For a more detailed explanation of the formalisation and the proof methods, see the accompanying paper (publication forthcoming).

Contents

1 Auxiliary lemmas

theory *Akra-Bazzi-Library*
imports
  Complex-Main
  Landau-Symbols.Landau-More
  Pure−ex.Guess
begin

lemma *ln-mono*: $0 < x \Longrightarrow 0 < y \Longrightarrow x \leq y \Longrightarrow ln\ (x::real) \leq ln\ y$
  ⟨*proof*⟩

lemma *ln-mono-strict*: $0 < x \Longrightarrow 0 < y \Longrightarrow x < y \Longrightarrow ln\ (x::real) < ln\ y$
  ⟨*proof*⟩

declare *DERIV-powr*[*THEN DERIV-chain2*, *derivative-intros*]

lemma *sum-pos'*:
  assumes *finite I*
  assumes $\exists x \in I.\ f\ x > (0 :: - :: linordered\text{-}ab\text{-}group\text{-}add)$
  assumes $\bigwedge x.\ x \in I \Longrightarrow f\ x \geq 0$
  shows $sum\ f\ I > 0$
⟨*proof*⟩

lemma *min-mult-left*:
  assumes $(x::real) > 0$
  shows $x * min\ y\ z = min\ (x*y)\ (x*z)$
  ⟨*proof*⟩

lemma *max-mult-left*:
  assumes $(x::real) > 0$
  shows $x * max\ y\ z = max\ (x*y)\ (x*z)$
  ⟨*proof*⟩

lemma *DERIV-nonneg-imp-mono*:
  assumes $\bigwedge t.\ t \in \{x..y\} \Longrightarrow (f\ has\text{-}field\text{-}derivative\ f'\ t)\ (at\ t)$
  assumes $\bigwedge t.\ t \in \{x..y\} \Longrightarrow f'\ t \geq 0$
  assumes $(x::real) \leq y$
  shows $(f\ x :: real) \leq f\ y$

⟨*proof*⟩

lemma *eventually-conjE*: *eventually* $(\lambda x.\ P\ x \wedge Q\ x)\ F \Longrightarrow$ (*eventually* $P\ F \Longrightarrow$ *eventually* $Q\ F \Longrightarrow R) \Longrightarrow R$
⟨*proof*⟩

lemma *real-natfloor-nat*: $x \in \mathbb{N} \Longrightarrow$ *real* (*nat* $\lfloor x \rfloor) = x$ ⟨*proof*⟩

lemma *eventually-natfloor*:
assumes *eventually* P (*at-top* :: *nat filter*)
shows *eventually* $(\lambda x.\ P$ (*nat* $\lfloor x \rfloor$)) (*at-top* :: *real filter*)
⟨*proof*⟩

lemma *tendsto-0-smallo-1*: $f \in o(\lambda x.\ 1 :: real) \Longrightarrow (f \longrightarrow 0)$ *at-top*
⟨*proof*⟩

lemma *smallo-1-tendsto-0*: $(f \longrightarrow 0)$ *at-top* $\Longrightarrow f \in o(\lambda x.\ 1 :: real)$
⟨*proof*⟩

lemma *filterlim-at-top-smallomega-1*:
assumes $f \in \omega[F](\lambda x.\ 1 :: real)$ *eventually* $(\lambda x.\ f\ x > 0)\ F$
shows *filterlim f at-top F*
⟨*proof*⟩

lemma *smallo-imp-abs-less-real*:
assumes $f \in o[F](g)$ *eventually* $(\lambda x.\ g\ x > (0::real))\ F$
shows *eventually* $(\lambda x.\ |f\ x| < g\ x)\ F$
⟨*proof*⟩

lemma *smallo-imp-less-real*:
assumes $f \in o[F](g)$ *eventually* $(\lambda x.\ g\ x > (0::real))\ F$
shows *eventually* $(\lambda x.\ f\ x < g\ x)\ F$
⟨*proof*⟩

lemma *smallo-imp-le-real*:
assumes $f \in o[F](g)$ *eventually* $(\lambda x.\ g\ x \geq (0::real))\ F$
shows *eventually* $(\lambda x.\ f\ x \leq g\ x)\ F$
⟨*proof*⟩

lemma *filterlim-at-right*:
filterlim f (*at-right a*) $F \longleftrightarrow$ *eventually* $(\lambda x.\ f\ x > a)\ F \wedge$ *filterlim f* (*nhds a*) F
⟨*proof*⟩

lemma *one-plus-x-powr-approx-ex*:
assumes x: *abs* $(x::real) \leq 1/2$
obtains t **where** *abs* $t < 1/2$ $(1 + x)$ *powr* $p =$
$1 + p * x + p * (p - 1) * (1 + t)$ *powr* $(p - 2) / 2 * x$ ^ 2

⟨*proof*⟩

lemma *powr-lower-bound*: $[\![(l::real) > 0; l \le x; x \le u]\!] \Longrightarrow min\ (l\ powr\ z)\ (u\ powr\ z) \le x\ powr\ z$
⟨*proof*⟩

lemma *powr-upper-bound*: $[\![(l::real) > 0; l \le x; x \le u]\!] \Longrightarrow max\ (l\ powr\ z)\ (u\ powr\ z) \ge x\ powr\ z$
⟨*proof*⟩

lemma *one-plus-x-powr-Taylor2*:
obtains *k* **where** $\bigwedge x.\ abs\ (x::real) \le 1/2 \Longrightarrow abs\ ((1 + x)\ powr\ p - 1 - p*x) \le k*x\hat{}2$
⟨*proof*⟩

lemma *one-plus-x-powr-Taylor2-bigo*:
assumes *lim*: $(f \longrightarrow 0)\ F$
shows $(\lambda x.\ (1 + f\ x)\ powr\ (p::real) - 1 - p * f\ x) \in O[F](\lambda x.\ f\ x\ \hat{}\ 2)$
⟨*proof*⟩

lemma *one-plus-x-powr-Taylor1-bigo*:
assumes *lim*: $(f \longrightarrow 0)\ F$
shows $(\lambda x.\ (1 + f\ x)\ powr\ (p::real) - 1) \in O[F](\lambda x.\ f\ x)$
⟨*proof*⟩

lemma *x-times-x-minus-1-nonneg*: $x \le 0 \lor x \ge 1 \Longrightarrow (x::\text{-}::linordered\text{-}idom) * (x - 1) \ge 0$
⟨*proof*⟩

lemma *x-times-x-minus-1-nonpos*: $x \ge 0 \Longrightarrow x \le 1 \Longrightarrow (x::\text{-}::linordered\text{-}idom) * (x - 1) \le 0$
⟨*proof*⟩

lemma *real-powr-at-bot*:
assumes $(a::real) > 1$
shows $((\lambda x.\ a\ powr\ x) \longrightarrow 0)\ at\text{-}bot$
⟨*proof*⟩

lemma *real-powr-at-bot-neg*:
assumes $(a::real) > 0\ a < 1$
shows *filterlim* $(\lambda x.\ a\ powr\ x)$ *at-top at-bot*
⟨*proof*⟩

lemma *real-powr-at-top-neg*:
assumes $(a::real) > 0\ a < 1$
shows $((\lambda x.\ a\ powr\ x) \longrightarrow 0)\ at\text{-}top$
⟨*proof*⟩

lemma *eventually-nat-real*:

assumes *eventually P* (*at-top* :: *real filter*)
shows *eventually* (λx. *P* (*real x*)) (*at-top* :: *nat filter*)
⟨*proof*⟩

end

2 Asymptotic bounds

theory *Akra-Bazzi-Asymptotics*
imports
Complex-Main
Akra-Bazzi-Library
HOL−Library.Landau-Symbols
begin

locale *akra-bazzi-asymptotics-bep* =
fixes *b e p hb* :: *real*
assumes *bep*: $b > 0$ $b < 1$ $e > 0$ $hb > 0$
begin

context
begin

Functions that are negligible w.r.t. *ln* ($b * x$) *powr* (e / 2 + 1).

private abbreviation (*input*) *negl* :: (*real* ⇒ *real*) ⇒ *bool* **where**
negl f ≡ *f* ∈ *o*(λx. *ln* ($b*x$) *powr* ($-(e/2 + 1)$))

private lemma *neglD*: *negl f* ⟹ $c > 0$ ⟹ *eventually* (λx. $|f\ x| \leq c$ / *ln* ($b*x$) *powr* ($e/2+1$)) *at-top*
⟨*proof*⟩ **lemma** *negl-mult*: *negl f* ⟹ *negl g* ⟹ *negl* (λx. $f\ x * g\ x$)
⟨*proof*⟩ **lemma** *ev4*:
assumes *g*: *negl g*
shows *eventually* (λx. *ln* ($b*x$) *powr* ($-e/2$) − *ln x powr* ($-e/2$) ≥ *g x*) *at-top*
⟨*proof*⟩ **lemma** *ev1*:
negl (λx. (1 + c * *inverse b* * *ln x powr* ($-(1+e)$)) *powr p* − 1)
⟨*proof*⟩ **lemma** *ev2-aux*:
defines *f* ≡ λx. (1 + 1/*ln* ($b*x$) * *ln* (1 + *hb* / *b* * *ln x powr* ($-1-e$))) *powr* ($-e/2$)
obtains *h* **where** *eventually* (λx. *f x* ≥ 1 + *h x*) *at-top* *h* ∈ *o*(λx. 1 / *ln x*)
⟨*proof*⟩ **lemma** *ev2*:
defines *f* ≡ λx. *ln* ($b * x$ + *hb* * *x* / *ln x powr* ($1 + e$)) *powr* ($-e/2$)
obtains *h* **where**
negl h
eventually (λx. *f x* ≥ *ln* ($b * x$) *powr* ($-e/2$) + *h x*) *at-top*
eventually (λx. |*ln* ($b * x$) *powr* ($-e/2$) + *h x*| < 1) *at-top*
⟨*proof*⟩ **lemma** *ev21*:
obtains *g* **where**
negl g
eventually (λx. 1 + *ln* ($b * x$ + *hb* * *x* / *ln x powr* ($1 + e$)) *powr* ($-e/2$) ≥

1 + ln (b ∗ x) powr (−e/2) + g x) at-top
eventually (λx. 1 + ln (b ∗ x) powr (−e/2) + g x > 0) at-top
⟨*proof*⟩ **lemma** *ev22*:
obtains *g* **where**
negl g
eventually (λx. 1 − ln (b ∗ x + hb ∗ x / ln x powr (1 + e)) powr (−e/2) ≤
1 − ln (b ∗ x) powr (−e/2) − g x) at-top
eventually (λx. 1 − ln (b ∗ x) powr (−e/2) − g x > 0) at-top
⟨*proof*⟩

lemma *asymptotics1*:
shows *eventually (λx.*
(1 + c ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗
(1 + ln (b ∗ x + hb ∗ x / ln x powr (1 + e)) powr (− e / 2)) ≥
1 + (ln x powr (−e/2))) at-top
⟨*proof*⟩

lemma *asymptotics2*:
shows *eventually (λx.*
(1 + c ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗
(1 − ln (b ∗ x + hb ∗ x / ln x powr (1 + e)) powr (− e / 2)) ≤
1 − (ln x powr (−e/2))) at-top
⟨*proof*⟩

lemma *asymptotics3*: *eventually (λx. (1 + (ln x powr (−e/2))) / 2 ≤ 1) at-top*
(**is** *eventually (λx. ?f x ≤ 1)* -)
⟨*proof*⟩

lemma *asymptotics4*: *eventually (λx. (1 − (ln x powr (−e/2))) ∗ 2 ≥ 1) at-top*
(**is** *eventually (λx. ?f x ≥ 1)* -)
⟨*proof*⟩

lemma *asymptotics5*: *eventually (λx. ln (b∗x − hb∗x∗ln x powr −(1+e)) powr (−e/2) < 1) at-top*
⟨*proof*⟩

lemma *asymptotics6*: *eventually (λx. hb / ln x powr (1 + e) < b/2) at-top*
and *asymptotics7*: *eventually (λx. hb / ln x powr (1 + e) < (1 − b) / 2) at-top*
and *asymptotics8*: *eventually (λx. x∗(1 − b − hb / ln x powr (1 + e)) > 1) at-top*
⟨*proof*⟩

end
end

definition *akra-bazzi-asymptotic1 b hb e p x ⟷*
(1 − hb ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗ (1 + ln (b∗x + hb∗x/ln x

powr (1+e)) powr (−e/2))
≥ 1 + (ln x powr (−e/2) :: real)
definition *akra-bazzi-asymptotic1′ b hb e p x ⟷*
(1 + hb ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗ (1 + ln (b∗x + hb∗x/ln x powr (1+e)) powr (−e/2))
≥ 1 + (ln x powr (−e/2) :: real)
definition *akra-bazzi-asymptotic2 b hb e p x ⟷*
(1 + hb ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗ (1 − ln (b∗x + hb∗x/ln x powr (1+e)) powr (−e/2))
≤ 1 − ln x powr (−e/2 :: real)
definition *akra-bazzi-asymptotic2′ b hb e p x ⟷*
(1 − hb ∗ inverse b ∗ ln x powr −(1+e)) powr p ∗ (1 − ln (b∗x + hb∗x/ln x powr (1+e)) powr (−e/2))
≤ 1 − ln x powr (−e/2 :: real)
definition *akra-bazzi-asymptotic3 e x ⟷ (1 + (ln x powr (−e/2))) / 2 ≤ (1::real)*
definition *akra-bazzi-asymptotic4 e x ⟷ (1 − (ln x powr (−e/2))) ∗ 2 ≥ (1::real)*
definition *akra-bazzi-asymptotic5 b hb e x ⟷*
ln (b∗x − hb∗x∗ln x powr −(1+e)) powr (−e/2::real) < 1

definition *akra-bazzi-asymptotic6 b hb e x ⟷ hb / ln x powr (1 + e :: real) < b/2*
definition *akra-bazzi-asymptotic7 b hb e x ⟷ hb / ln x powr (1 + e :: real) < (1 − b) / 2*
definition *akra-bazzi-asymptotic8 b hb e x ⟷ x∗(1 − b − hb / ln x powr (1 + e :: real)) > 1*

definition *akra-bazzi-asymptotics b hb e p x ⟷*
akra-bazzi-asymptotic1 b hb e p x ∧ akra-bazzi-asymptotic1′ b hb e p x ∧
akra-bazzi-asymptotic2 b hb e p x ∧ akra-bazzi-asymptotic2′ b hb e p x ∧
akra-bazzi-asymptotic3 e x ∧ akra-bazzi-asymptotic4 e x ∧ akra-bazzi-asymptotic5 b hb e x ∧
akra-bazzi-asymptotic6 b hb e x ∧ akra-bazzi-asymptotic7 b hb e x ∧
akra-bazzi-asymptotic8 b hb e x

lemmas *akra-bazzi-asymptotic-defs =*
akra-bazzi-asymptotic1-def akra-bazzi-asymptotic1′-def
akra-bazzi-asymptotic2-def akra-bazzi-asymptotic2′-def akra-bazzi-asymptotic3-def
akra-bazzi-asymptotic4-def akra-bazzi-asymptotic5-def akra-bazzi-asymptotic6-def
akra-bazzi-asymptotic7-def akra-bazzi-asymptotic8-def akra-bazzi-asymptotics-def

lemma *akra-bazzi-asymptotics*:
assumes *⋀b. b ∈ set bs ⟹ b ∈ {0<..<1}*
assumes *hb > 0 e > 0*
shows *eventually (λx. ∀ b∈set bs. akra-bazzi-asymptotics b hb e p x) at-top*
⟨*proof*⟩

end

3 The continuous Akra-Bazzi theorem

theory *Akra-Bazzi-Real*
imports
  Complex-Main
  Akra-Bazzi-Asymptotics
begin

We want to be generic over the integral definition used; we fix some arbitrary notions of integrability and integral and assume just the properties we need. The user can then instantiate the theorems with any desired integral definition.

locale *akra-bazzi-integral* =
  fixes *integrable* :: (*real* ⇒ *real*) ⇒ *real* ⇒ *real* ⇒ *bool*
    and *integral* :: (*real* ⇒ *real*) ⇒ *real* ⇒ *real* ⇒ *real*
  assumes *integrable-const*: $c \geq 0 \Longrightarrow$ *integrable* (λ-. *c*) *a b*
      and *integral-const*: $c \geq 0 \Longrightarrow a \leq b \Longrightarrow$ *integral* (λ-. *c*) *a b* = (*b* − *a*) ∗ *c*
      and *integrable-subinterval*:
          integrable f a b ⟹ $a \leq a' \Longrightarrow b' \leq b \Longrightarrow$ *integrable f a′ b′*
      and *integral-le*:
          integrable f a b ⟹ *integrable g a b* ⟹ (⋀*x*. *x* ∈ {*a*..*b*} ⟹ *f x* ≤ *g x*) ⟹
              integral f a b ≤ *integral g a b*
      and *integral-combine*:
          a ≤ *c* ⟹ *c* ≤ *b* ⟹ *integrable f a b* ⟹
              integral f a c + *integral f c b* = *integral f a b*
begin
lemma *integral-nonneg*:
  a ≤ *b* ⟹ *integrable f a b* ⟹ (⋀*x*. *x* ∈ {*a*..*b*} ⟹ *f x* ≥ *0*) ⟹ *integral f a b* ≥ *0*
  ⟨*proof*⟩
end

declare *sum.cong*[*fundef-cong*]

lemma *strict-mono-imp-ex1-real*:
  fixes *f* :: *real* ⇒ *real*
  assumes *lim-neg-inf*: *LIM x at-bot. f x* :> *at-top*
  assumes *lim-inf*: (*f* ⟶ *z*) *at-top*
  assumes *mono*: ⋀*a b*. *a* < *b* ⟹ *f b* < *f a*
  assumes *cont*: ⋀*x*. *isCont f x*
  assumes *y-greater-z*: *z* < *y*
  shows ∃!*x*. *f x* = *y*
⟨*proof*⟩

The parameter *p* in the Akra-Bazzi theorem always exists and is unique.

definition *akra-bazzi-exponent* :: *real list* ⇒ *real list* ⇒ *real* **where**
  akra-bazzi-exponent as bs ≡ (*THE p*. (∑ *i*<*length as*. *as*!*i* ∗ *bs*!*i powr p*) = *1*)

locale *akra-bazzi-params* =
  fixes k :: *nat* **and** *as bs* :: *real list*
  assumes *length-as*: *length as* = k
  and *length-bs*: *length bs* = k
  and *k-not-0*: $k \neq 0$
  and *a-ge-0*: $a \in$ *set as* $\Longrightarrow a \geq 0$
  and *b-bounds*: $b \in$ *set bs* $\Longrightarrow b \in \{0<..<1\}$
begin

abbreviation p :: *real* **where** $p \equiv$ *akra-bazzi-exponent as bs*

lemma *p-def*: $p = (THE\ p.\ (\sum i<k.\ as!i * bs!i\ powr\ p) = 1)$
  ⟨*proof*⟩

lemma *b-pos*: $b \in$ *set bs* $\Longrightarrow b > 0$ **and** *b-less-1*: $b \in$ *set bs* $\Longrightarrow b < 1$
  ⟨*proof*⟩

lemma *as-nonempty* [*simp*]: *as* $\neq$ [] **and** *bs-nonempty* [*simp*]: *bs* $\neq$ []
  ⟨*proof*⟩

lemma *a-in-as*[*intro*, *simp*]: $i < k \Longrightarrow$ *as* ! $i \in$ *set as*
  ⟨*proof*⟩

lemma *b-in-bs*[*intro*, *simp*]: $i < k \Longrightarrow$ *bs* ! $i \in$ *set bs*
  ⟨*proof*⟩

end

locale *akra-bazzi-params-nonzero* =
  fixes k :: *nat* **and** *as bs* :: *real list*
  assumes *length-as*: *length as* = k
  and *length-bs*: *length bs* = k
  and *a-ge-0*: $a \in$ *set as* $\Longrightarrow a \geq 0$
  and *ex-a-pos*: $\exists a \in$ *set as*. $a > 0$
  and *b-bounds*: $b \in$ *set bs* $\Longrightarrow b \in \{0<..<1\}$
begin

sublocale *akra-bazzi-params k as bs*
⟨*proof*⟩

lemma *akra-bazzi-p-strict-mono*:
  assumes $x < y$
  shows $(\sum i<k.\ as!i * bs!i\ powr\ y) < (\sum i<k.\ as!i * bs!i\ powr\ x)$
⟨*proof*⟩

lemma *akra-bazzi-p-mono*:
  assumes $x \leq y$

shows $(\sum i<k.\ as!i * bs!i\ powr\ y) \leq (\sum i<k.\ as!i * bs!i\ powr\ x)$
⟨*proof*⟩

lemma *akra-bazzi-p-unique*:
$\exists!p.\ (\sum i<k.\ as!i * bs!i\ powr\ p) = 1$
⟨*proof*⟩

lemma *p-props*: $(\sum i<k.\ as!i * bs!i\ powr\ p) = 1$
and *p-unique*: $(\sum i<k.\ as!i * bs!i\ powr\ p') = 1 \Longrightarrow p = p'$
⟨*proof*⟩

lemma *p-greaterI*: $1 < (\sum i<k.\ as!i * bs!i\ powr\ p') \Longrightarrow p' < p$
⟨*proof*⟩

lemma *p-lessI*: $1 > (\sum i<k.\ as!i * bs!i\ powr\ p') \Longrightarrow p' > p$
⟨*proof*⟩

lemma *p-geI*: $1 \leq (\sum i<k.\ as!i * bs!i\ powr\ p') \Longrightarrow p' \leq p$
⟨*proof*⟩

lemma *p-leI*: $1 \geq (\sum i<k.\ as!i * bs!i\ powr\ p') \Longrightarrow p' \geq p$
⟨*proof*⟩

lemma *p-boundsI*: $(\sum i<k.\ as!i * bs!i\ powr\ x) \leq 1 \land (\sum i<k.\ as!i * bs!i\ powr\ y) \geq 1 \Longrightarrow p \in \{y..x\}$
⟨*proof*⟩

lemma *p-boundsI'*: $(\sum i<k.\ as!i * bs!i\ powr\ x) < 1 \land (\sum i<k.\ as!i * bs!i\ powr\ y) > 1 \Longrightarrow p \in \{y<..<x\}$
⟨*proof*⟩

lemma *p-nonneg*: *sum-list as* $\geq 1 \Longrightarrow p \geq 0$
⟨*proof*⟩

end

locale *akra-bazzi-real-recursion* =
fixes *as bs* :: *real list* **and** *hs* :: (*real* ⇒ *real*) *list* **and** *k* :: *nat* **and** x_0 x_1 *hb e p* :: *real*
assumes *length-as*: *length as* = *k*
and *length-bs*: *length bs* = *k*
and *length-hs*: *length hs* = *k*
and *k-not-0*: $k \neq 0$
and *a-ge-0*: $a \in set\ as \Longrightarrow a \geq 0$
and *b-bounds*: $b \in set\ bs \Longrightarrow b \in \{0<..<1\}$

and *x0-ge-1*: $x_0 \geq 1$
and *x0-le-x1*: $x_0 \leq x_1$
and *x1-ge*: $b \in set\ bs \Longrightarrow x_1 \geq 2 * x_0 * inverse\ b$

and *e-pos*: $e > 0$
and *h-bounds*: $x \geq x_1 \Longrightarrow h \in set\ hs \Longrightarrow |h\ x| \leq hb * x\ /\ ln\ x\ powr\ (1 + e)$

and *asymptotics*: $x \geq x_0 \Longrightarrow b \in set\ bs \Longrightarrow akra\text{-}bazzi\text{-}asymptotics\ b\ hb\ e\ p\ x$
begin

sublocale *akra-bazzi-params k as bs*
⟨*proof*⟩

lemma *h-in-hs*[*intro*, *simp*]: $i < k \Longrightarrow hs\ !\ i \in set\ hs$
⟨*proof*⟩

lemma *x1-gt-1*: $x_1 > 1$
⟨*proof*⟩

lemma *x1-ge-1*: $x_1 \geq 1$ ⟨*proof*⟩

lemma *x1-pos*: $x_1 > 0$ ⟨*proof*⟩

lemma *bx-le-x*: $x \geq 0 \Longrightarrow b \in set\ bs \Longrightarrow b * x \leq x$
⟨*proof*⟩

lemma *x0-pos*: $x_0 > 0$ ⟨*proof*⟩

lemma
assumes $x \geq x_0$ $b \in set\ bs$
shows *x0-hb-bound0*: $hb\ /\ ln\ x\ powr\ (1 + e) < b/2$
and *x0-hb-bound1*: $hb\ /\ ln\ x\ powr\ (1 + e) < (1 - b)\ /\ 2$
and *x0-hb-bound2*: $x*(1 - b - hb\ /\ ln\ x\ powr\ (1 + e)) > 1$
⟨*proof*⟩

lemma *step-diff*:
assumes $i < k$ $x \geq x_1$
shows $bs\ !\ i * x + (hs\ !\ i)\ x + 1 < x$
⟨*proof*⟩

lemma *step-le-x*: $i < k \Longrightarrow x \geq x_1 \Longrightarrow bs\ !\ i * x + (hs\ !\ i)\ x \leq x$
⟨*proof*⟩

lemma *x0-hb-bound0'*: $\bigwedge x\ b.\ x \geq x_0 \Longrightarrow b \in set\ bs \Longrightarrow hb\ /\ ln\ x\ powr\ (1 + e) < b$
⟨*proof*⟩

lemma *step-pos*:

assumes $i < k$ $x \geq x_1$
shows $bs\ !\ i * x + (hs\ !\ i)\ x > 0$
⟨*proof*⟩

lemma *step-nonneg*: $i < k \Longrightarrow x \geq x_1 \Longrightarrow bs\ !\ i * x + (hs\ !\ i)\ x \geq 0$
⟨*proof*⟩

lemma *step-nonneg'*: $i < k \Longrightarrow x \geq x_1 \Longrightarrow bs\ !\ i + (hs\ !\ i)\ x\ /\ x \geq 0$
⟨*proof*⟩

lemma *hb-nonneg*: $hb \geq 0$
⟨*proof*⟩

lemma *x0-hb-bound3*:
assumes $x \geq x_1$ $i < k$
shows $x - (bs\ !\ i * x + (hs\ !\ i)\ x) \leq x$
⟨*proof*⟩

lemma *x0-hb-bound4*:
assumes $x \geq x_1$ $i < k$
shows $(bs\ !\ i + (hs\ !\ i)\ x\ /\ x) > bs\ !\ i\ /\ 2$
⟨*proof*⟩

lemma *x0-hb-bound4'*: $x \geq x_1 \Longrightarrow i < k \Longrightarrow (bs\ !\ i + (hs\ !\ i)\ x\ /\ x) \geq bs\ !\ i\ /\ 2$
⟨*proof*⟩

lemma *x0-hb-bound5*:
assumes $x \geq x_1$ $i < k$
shows $(bs\ !\ i + (hs\ !\ i)\ x\ /\ x) \leq bs\ !\ i * 3/2$
⟨*proof*⟩

lemma *x0-hb-bound6*:
assumes $x \geq x_1$ $i < k$
shows $x * ((1 - bs\ !\ i)\ /\ 2) \leq x - (bs\ !\ i * x + (hs\ !\ i)\ x)$
⟨*proof*⟩

lemma *x0-hb-bound7*:
assumes $x \geq x_1$ $i < k$
shows $bs!i*x + (hs!i)\ x > x_0$
⟨*proof*⟩

lemma *x0-hb-bound7'*: $x \geq x_1 \Longrightarrow i < k \Longrightarrow bs!i*x + (hs!i)\ x > 1$
⟨*proof*⟩

lemma *x0-hb-bound8*:
assumes $x \geq x_1$ $i < k$
shows $bs!i*x - hb * x\ /\ ln\ x\ powr\ (1+e) > x_0$
⟨*proof*⟩

lemma *x0-hb-bound8'*:
  assumes $x \geq x_1$ $i < k$
  shows $bs!i*x + hb * x / ln\ x\ powr\ (1+e) > x_0$
⟨*proof*⟩

lemma
  assumes $x \geq x_0$
  shows *asymptotics1*: $i < k \Longrightarrow 1 + ln\ x\ powr\ (-\ e\ /\ 2) \leq$
    $(1 - hb * inverse\ (bs!i) * ln\ x\ powr\ -(1+e))\ powr\ p *$
    $(1 + ln\ (bs!i*x + hb*x/ln\ x\ powr\ (1+e))\ powr\ (-e/2))$
  and *asymptotics2*: $i < k \Longrightarrow 1 - ln\ x\ powr\ (-\ e\ /\ 2) \geq$
    $(1 + hb * inverse\ (bs!i) * ln\ x\ powr\ -(1+e))\ powr\ p *$
    $(1 - ln\ (bs!i*x + hb*x/ln\ x\ powr\ (1+e))\ powr\ (-e/2))$
  and *asymptotics1'*: $i < k \Longrightarrow 1 + ln\ x\ powr\ (-\ e\ /\ 2) \leq$
    $(1 + hb * inverse\ (bs!i) * ln\ x\ powr\ -(1+e))\ powr\ p *$
    $(1 + ln\ (bs!i*x + hb*x/ln\ x\ powr\ (1+e))\ powr\ (-e/2))$
  and *asymptotics2'*: $i < k \Longrightarrow 1 - ln\ x\ powr\ (-\ e\ /\ 2) \geq$
    $(1 - hb * inverse\ (bs!i) * ln\ x\ powr\ -(1+e))\ powr\ p *$
    $(1 - ln\ (bs!i*x + hb*x/ln\ x\ powr\ (1+e))\ powr\ (-e/2))$
  and *asymptotics3*: $(1 + ln\ x\ powr\ (-\ e\ /\ 2))\ /\ 2 \leq 1$
  and *asymptotics4*: $(1 - ln\ x\ powr\ (-\ e\ /\ 2)) * 2 \geq 1$
  and *asymptotics5*: $i < k \Longrightarrow ln\ (bs!i*x - hb*x*ln\ x\ powr\ -(1+e))\ powr$
$(-e/2) < 1$
⟨*proof*⟩

lemma *x0-hb-bound9*:
  assumes $x \geq x_1$ $i < k$
  shows $ln\ (bs!i*x + (hs!i)\ x)\ powr\ -(e/2) < 1$
⟨*proof*⟩

definition *akra-bazzi-measure* :: *real* ⇒ *nat* **where**
  akra-bazzi-measure $x = nat\ \lceil x \rceil$

lemma *akra-bazzi-measure-decreases*:
  assumes $x \geq x_1$ $i < k$
  shows *akra-bazzi-measure* $(bs!i*x + (hs!i)\ x) <$ *akra-bazzi-measure* x
⟨*proof*⟩

lemma *akra-bazzi-induct*[*consumes 1*, *case-names base rec*]:
  assumes $x \geq x_0$
  assumes *base*: $\bigwedge x.\ x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow P\ x$
  assumes *rec*: $\bigwedge x.\ x > x_1 \Longrightarrow (\bigwedge i.\ i < k \Longrightarrow P\ (bs!i*x + (hs!i)\ x)) \Longrightarrow P\ x$
  shows $P\ x$
⟨*proof*⟩

end

locale *akra-bazzi-real* = *akra-bazzi-real-recursion* +
fixes *integrable integral*
assumes *integral*: *akra-bazzi-integral integrable integral*
fixes f :: *real* ⇒ *real*
and g :: *real* ⇒ *real*
and C :: *real*
assumes *p-props*: $(\sum i<k.\ as!i * bs!i \ powr\ p) = 1$
and *f-base*: $x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f\ x \geq 0$
and *f-rec*: $x > x_1 \Longrightarrow f\ x = g\ x + (\sum i<k.\ as!i * f\ (bs!i * x + (hs!i)\ x))$
and *g-nonneg*: $x \geq x_0 \Longrightarrow g\ x \geq 0$
and *C-bound*: $b \in set\ bs \Longrightarrow x \geq x_1 \Longrightarrow C*x \leq b*x - hb*x/ln\ x\ powr\ (1+e)$
and *g-integrable*: $x \geq x_0 \Longrightarrow integrable\ (\lambda u.\ g\ u\ /\ u\ powr\ (p + 1))\ x_0\ x$
begin

interpretation *akra-bazzi-integral integrable integral* ⟨*proof*⟩

lemma *akra-bazzi-integrable*:
$a \geq x_0 \Longrightarrow a \leq b \Longrightarrow integrable\ (\lambda x.\ g\ x\ /\ x\ powr\ (p + 1))\ a\ b$
⟨*proof*⟩

definition *g-approx* :: *nat* ⇒ *real* ⇒ *real* **where**
$g\text{-}approx\ i\ x = x\ powr\ p * integral\ (\lambda u.\ g\ u\ /\ u\ powr\ (p + 1))\ (bs!i * x + (hs!i)\ x)\ x$

lemma *f-nonneg*: $x \geq x_0 \Longrightarrow f\ x \geq 0$
⟨*proof*⟩

definition *f-approx* :: *real* ⇒ *real* **where**
$f\text{-}approx\ x = x\ powr\ p * (1 + integral\ (\lambda u.\ g\ u\ /\ u\ powr\ (p + 1))\ x_0\ x)$

lemma *f-approx-aux*:
assumes $x \geq x_0$
shows $1 + integral\ (\lambda u.\ g\ u\ /\ u\ powr\ (p + 1))\ x_0\ x \geq 1$
⟨*proof*⟩

lemma *f-approx-pos*: $x \geq x_0 \Longrightarrow f\text{-}approx\ x > 0$
⟨*proof*⟩

lemma *f-approx-nonneg*: $x \geq x_0 \Longrightarrow f\text{-}approx\ x \geq 0$
⟨*proof*⟩

lemma *f-approx-bounded-below*:
obtains c **where** $\bigwedge x.\ x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f\text{-}approx\ x \geq c\ \ c > 0$

⟨*proof*⟩

lemma *asymptotics-aux*:
  assumes $x \geq x_1$ $i < k$
  assumes $s \equiv (if\ p \geq 0\ then\ 1\ else\ -1)$
  shows $(bs!i*x - s*hb*x*ln\ x\ powr\ -(1+e))\ powr\ p \leq (bs!i*x + (hs!i)\ x)\ powr\ p$ (**is** *?thesis1*)
  and $(bs!i*x + (hs!i)\ x)\ powr\ p \leq (bs!i*x + s*hb*x*ln\ x\ powr\ -(1+e))\ powr\ p$ (**is** *?thesis2*)
⟨*proof*⟩

lemma *asymptotics1'*:
  assumes $x \geq x_1$ $i < k$
  shows $(bs!i*x)\ powr\ p * (1 + ln\ x\ powr\ (-e/2)) \leq$
    $(bs!i*x + (hs!i)\ x)\ powr\ p * (1 + ln\ (bs!i*x + (hs!i)\ x)\ powr\ (-e/2))$
⟨*proof*⟩

lemma *asymptotics2'*:
  assumes $x \geq x_1$ $i < k$
  shows $(bs!i*x + (hs!i)\ x)\ powr\ p * (1 - ln\ (bs!i*x + (hs!i)\ x)\ powr\ (-e/2)) \leq$
    $(bs!i*x)\ powr\ p * (1 - ln\ x\ powr\ (-e/2))$
⟨*proof*⟩

lemma *Cx-le-step*:
  assumes $i < k$ $x \geq x_1$
  shows $C*x \leq bs!i*x + (hs!i)\ x$
⟨*proof*⟩

end

locale *akra-bazzi-nat-to-real* = *akra-bazzi-real-recursion* +
  fixes $f :: nat \Rightarrow real$
  and $g :: real \Rightarrow real$
  assumes *f-base*: $real\ x \geq x_0 \Longrightarrow real\ x \leq x_1 \Longrightarrow f\ x \geq 0$
  and *f-rec*: $real\ x > x_1 \Longrightarrow$
    $f\ x = g\ (real\ x) + (\sum i{<}k.\ as!i * f\ (nat\ \lfloor bs!i * x + (hs!i)\ (real\ x)\rfloor))$
  and *x0-int*: $real\ (nat\ \lfloor x_0\rfloor) = x_0$
begin

function $f' :: real \Rightarrow real$ **where**
  $x \leq x_1 \Longrightarrow f'\ x = f\ (nat\ \lfloor x\rfloor)$
| $x > x_1 \Longrightarrow f'\ x = g\ x + (\sum i{<}k.\ as!i * f'\ (bs!i * x + (hs!i)\ x))$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *f'-base*: $x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f' \; x \geq 0$
⟨*proof*⟩

lemmas *f'-rec* = *f'.simps(2)*

end

locale *akra-bazzi-real-lower* = *akra-bazzi-real* +
fixes *fb2 gb2 c2* :: *real*
assumes *f-base2*: $x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f \; x \geq fb2$
and *fb2-pos*: $fb2 > 0$
and *g-growth2*: $\forall x \geq x_1. \; \forall u \in \{C*x..x\}. \; c2 * g \; x \geq g \; u$
and *c2-pos*: $c2 > 0$
and *g-bounded*: $x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow g \; x \leq gb2$
begin

interpretation *akra-bazzi-integral integrable integral* ⟨*proof*⟩

lemma *gb2-nonneg*: $gb2 \geq 0$ ⟨*proof*⟩

lemma *g-growth2'*:
assumes $x \geq x_1 \; i < k \; u \in \{bs!i*x+(hs!i) \; x..x\}$
shows $c2 * g \; x \geq g \; u$
⟨*proof*⟩

lemma *g-bounds2*:
obtains *c4* **where** $\bigwedge x \; i. \; x \geq x_1 \Longrightarrow i < k \Longrightarrow g\text{-}approx \; i \; x \leq c4 * g \; x \; c4 > 0$
⟨*proof*⟩

lemma *f-approx-bounded-above*:
obtains *c* **where** $\bigwedge x. \; x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f\text{-}approx \; x \leq c \; c > 0$
⟨*proof*⟩

lemma *f-bounded-below*:
assumes *c'*: $c' > 0$
obtains *c* **where** $\bigwedge x. \; x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow 2 * (c * f\text{-}approx \; x) \leq f \; x \; c \leq c'$
$c > 0$
⟨*proof*⟩

lemma *akra-bazzi-lower*:
obtains *c5* **where** $\bigwedge x. \; x \geq x_0 \Longrightarrow f \; x \geq c5 * f\text{-}approx \; x \; c5 > 0$
⟨*proof*⟩

lemma *akra-bazzi-bigomega*:
$f \in \Omega(\lambda x. \; x \; powr \; p * (1 + integral \; (\lambda u. \; g \; u \; / \; u \; powr \; (p + 1)) \; x_0 \; x))$
⟨*proof*⟩

end

locale *akra-bazzi-real-upper* = *akra-bazzi-real* +
fixes *fb1 c1* :: *real*
assumes *f-base1*: $x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f\ x \leq fb1$
and *g-growth1*: $\forall x \geq x_1.\ \forall u \in \{C*x..x\}.\ c1 * g\ x \leq g\ u$
and *c1-pos*: $c1 > 0$
begin

interpretation *akra-bazzi-integral integrable integral* ⟨*proof*⟩

lemma *g-growth1'*:
assumes $x \geq x_1$ $i < k$ $u \in \{bs!i*x+(hs!i)\ x..x\}$
shows $c1 * g\ x \leq g\ u$
⟨*proof*⟩

lemma *g-bounds1*:
obtains *c3* **where**
$\bigwedge x\ i.\ x \geq x_1 \Longrightarrow i < k \Longrightarrow c3 * g\ x \leq g\text{-}approx\ i\ x$ $c3 > 0$
⟨*proof*⟩

lemma *f-bounded-above*:
assumes c': $c' > 0$
obtains *c* **where** $\bigwedge x.\ x \geq x_0 \Longrightarrow x \leq x_1 \Longrightarrow f\ x \leq (1/2) * (c * f\text{-}approx\ x)$ $c \geq c'$ $c > 0$
⟨*proof*⟩

lemma *akra-bazzi-upper*:
obtains *c6* **where** $\bigwedge x.\ x \geq x_0 \Longrightarrow f\ x \leq c6 * f\text{-}approx\ x$ $c6 > 0$
⟨*proof*⟩

lemma *akra-bazzi-bigo*:
$f \in O(\lambda x.\ x\ powr\ p *(1 + integral\ (\lambda u.\ g\ u\ /\ u\ powr\ (p + 1))\ x_0\ x))$
⟨*proof*⟩

end

end

4 The discrete Akra-Bazzi theorem

theory *Akra-Bazzi*
imports
Complex-Main
HOL−Library.Landau-Symbols
Akra-Bazzi-Real
begin

lemma *ex-mono*: $(\exists\, x.\ P\ x) \Longrightarrow (\bigwedge x.\ P\ x \Longrightarrow Q\ x) \Longrightarrow (\exists\, x.\ Q\ x)$ ⟨*proof*⟩

lemma *x-over-ln-mono*:
assumes $(e{::}real) > 0$
assumes $x > exp\ e$
assumes $x \leq y$
shows $x\ /\ ln\ x\ powr\ e \leq y\ /\ ln\ y\ powr\ e$
⟨*proof*⟩

definition *akra-bazzi-term* :: $nat \Rightarrow nat \Rightarrow real \Rightarrow (nat \Rightarrow nat) \Rightarrow bool$ **where**
akra-bazzi-term $x_0\ x_1\ b\ t =$
$(\exists\, e\ h.\ e > 0 \wedge (\lambda x.\ h\ x) \in O(\lambda x.\ real\ x\ /\ ln\ (real\ x)\ powr\ (1{+}e)) \wedge$
$(\forall\, x{\geq}x_1.\ t\ x \geq x_0 \wedge t\ x < x \wedge b{*}x + h\ x = real\ (t\ x)))$

lemma *akra-bazzi-termI* [*intro?*]:
assumes $e > 0\ (\lambda x.\ h\ x) \in O(\lambda x.\ real\ x\ /\ ln\ (real\ x)\ powr\ (1{+}e))$
$\bigwedge x.\ x \geq x_1 \Longrightarrow t\ x \geq x_0\ \bigwedge x.\ x \geq x_1 \Longrightarrow t\ x < x$
$\bigwedge x.\ x \geq x_1 \Longrightarrow b{*}x + h\ x = real\ (t\ x)$
shows *akra-bazzi-term* $x_0\ x_1\ b\ t$
⟨*proof*⟩

lemma *akra-bazzi-term-imp-less*:
assumes *akra-bazzi-term* $x_0\ x_1\ b\ t\ x \geq x_1$
shows $t\ x < x$
⟨*proof*⟩

lemma *akra-bazzi-term-imp-less'*:
assumes *akra-bazzi-term* $x_0\ (Suc\ x_1)\ b\ t\ x > x_1$
shows $t\ x < x$
⟨*proof*⟩

locale *akra-bazzi-recursion* =
fixes $x_0\ x_1\ k$ :: *nat* **and** *as bs* :: *real list* **and** *ts* :: $(nat \Rightarrow nat)$ *list* **and** f :: $nat \Rightarrow real$
assumes *k-not-0*: $k \neq 0$
and *length-as*: $length\ as = k$
and *length-bs*: $length\ bs = k$
and *length-ts*: $length\ ts = k$
and *a-ge-0*: $a \in set\ as \Longrightarrow a \geq 0$
and *b-bounds*: $b \in set\ bs \Longrightarrow b \in \{0{<}..{<}1\}$
and *ts*: $i < length\ bs \Longrightarrow$ *akra-bazzi-term* $x_0\ x_1\ (bs!i)\ (ts!i)$
begin

sublocale *akra-bazzi-params k as bs*
⟨*proof*⟩

lemma *ts-nonempty*: $ts \neq []$ ⟨*proof*⟩

definition *e-hs* :: *real* × (*nat* ⇒ *real*) *list* **where**
e-hs = (*SOME* (*e*,*hs*).
e > *0* ∧ *length hs* = *k* ∧ (∀ *h*∈*set hs*. (λ*x*. *h x*) ∈ *O*(λ*x*. *real x* / *ln* (*real x*) *powr* (*1*+*e*))) ∧
(∀ *t*∈*set ts*. ∀ $x \geq x_1$. *t x* ≥ x_0 ∧ *t x* < *x*) ∧
(∀ *i*<*k*. ∀ $x \geq x_1$. (*bs*!*i*)∗*x* + (*hs*!*i*) *x* = *real* ((*ts*!*i*) *x*))
)

definition *e* = (*case e-hs of* (*e*,-) ⇒ *e*)
definition *hs* = (*case e-hs of* (-,*hs*) ⇒ *hs*)

lemma *filterlim-powr-zero-cong*:
filterlim (λ*x*. *P* (*x*::*real*) (*x powr* (*0*::*real*))) *F at-top* = *filterlim* (λ*x*. *P x 1*) *F at-top*
⟨*proof*⟩

lemma *e-hs-aux*:
0 < *e* ∧ *length hs* = *k* ∧
(∀ *h*∈*set hs*. (λ*x*. *h x*) ∈ *O*(λ*x*. *real x* / *ln* (*real x*) *powr* (*1* + *e*))) ∧
(∀ *t*∈*set ts*. ∀ $x \geq x_1$. x_0 ≤ *t x* ∧ *t x* < *x*) ∧
(∀ *i*<*k*. ∀ $x \geq x_1$. (*bs*!*i*)∗*x* + (*hs*!*i*) *x* = *real* ((*ts*!*i*) *x*))
⟨*proof*⟩

lemma
e-pos: *e* > *0* **and** *length-hs*: *length hs* = *k* **and**
hs-growth: ⋀*h*. *h* ∈ *set hs* ⟹ (λ*x*. *h x*) ∈ *O*(λ*x*. *real x* / *ln* (*real x*) *powr* (*1* + *e*)) **and**
step-ge-x0: ⋀*t x*. *t* ∈ *set ts* ⟹ $x \geq x_1$ ⟹ $x_0 \leq$ *t x* **and**
step-less: ⋀*t x*. *t* ∈ *set ts* ⟹ $x \geq x_1$ ⟹ *t x* < *x* **and**
decomp: ⋀*i x*. *i* < *k* ⟹ $x \geq x_1$ ⟹ (*bs*!*i*)∗*x* + (*hs*!*i*) *x* = *real* ((*ts*!*i*) *x*)
⟨*proof*⟩

lemma *h-in-hs* [*intro*, *simp*]: *i* < *k* ⟹ *hs* ! *i* ∈ *set hs*
⟨*proof*⟩

lemma *t-in-ts* [*intro*, *simp*]: *i* < *k* ⟹ *ts* ! *i* ∈ *set ts*
⟨*proof*⟩

lemma *x0-less-x1*: $x_0 < x_1$ **and** *x0-le-x1*: $x_0 \leq x_1$
⟨*proof*⟩

lemma *akra-bazzi-induct* [*consumes 1*, *case-names base rec*]:
assumes $x \geq x_0$
assumes *base*: ⋀*x*. $x \geq x_0$ ⟹ $x < x_1$ ⟹ *P x*
assumes *rec*: ⋀*x*. $x \geq x_1$ ⟹ (⋀*t*. *t* ∈ *set ts* ⟹ *P* (*t x*)) ⟹ *P x*
shows *P x*

⟨*proof*⟩

end

locale *akra-bazzi-function* = *akra-bazzi-recursion* +
  fixes *integrable integral*
  assumes *integral*: *akra-bazzi-integral integrable integral*
  fixes g :: *nat* ⇒ *real*
  assumes *f-nonneg-base*: $x \geq x_0 \Longrightarrow x < x_1 \Longrightarrow f\, x \geq 0$
  and *f-rec*: $x \geq x_1 \Longrightarrow f\, x = g\, x + (\sum i{<}k.\ as!i * f\ ((ts!i)\ x))$
  and *g-nonneg*: $x \geq x_1 \Longrightarrow g\ x \geq 0$
  and *ex-pos-a*: $\exists\, a{\in}set\ as.\ a > 0$
begin

lemma *ex-pos-a′*: $\exists\, i{<}k.\ as!i > 0$
  ⟨*proof*⟩

sublocale *akra-bazzi-params-nonzero*
  ⟨*proof*⟩

definition *g-real* :: *real* ⇒ *real* **where** *g-real* $x = g\ (nat\ \lfloor x \rfloor)$

lemma *g-real-real*[*simp*]: *g-real* (*real* x) = $g\ x$ ⟨*proof*⟩

lemma *f-nonneg*: $x \geq x_0 \Longrightarrow f\, x \geq 0$
⟨*proof*⟩

definition $hs' = map\ (\lambda h\ x.\ h\ (nat\ \lfloor x{::}real \rfloor))\ hs$

lemma *length-hs′*: *length* $hs' = k$ ⟨*proof*⟩

lemma *hs′-real*: $i < k \Longrightarrow (hs'!i)\ (real\ x) = (hs!i)\ x$
  ⟨*proof*⟩

lemma *h-bound*:
  obtains *hb* **where** $hb > 0$ **and**
    eventually $(\lambda x.\ \forall\, h{\in}set\ hs'.\ |h\ x| \leq hb * x\ /\ ln\ x\ powr\ (1 + e))$ *at-top*
⟨*proof*⟩

lemma *C-bound*:
  assumes $\bigwedge b.\ b \in set\ bs \Longrightarrow C < b\ hb > 0$
  shows *eventually* $(\lambda x{::}real.\ \forall\, b{\in}set\ bs.\ C{*}x \leq b{*}x - hb{*}x/ln\ x\ powr\ (1{+}e))$
at-top
⟨*proof*⟩

end

locale *akra-bazzi-lower* = *akra-bazzi-function* +
fixes g' :: *real* ⇒ *real*
assumes *f-pos*: *eventually* ($\lambda x.$ *f x* > *0*) *at-top*
and *g-growth2*: ∃ *C c2*. *c2* > *0* ∧ *C* < *Min* (*set bs*) ∧
eventually ($\lambda x.$ ∀ *u*∈{*C*∗*x*..*x*}. g' *u* ≤ *c2* ∗ g' *x*) *at-top*
and g'*-integrable*: ∃ *a*. ∀ *b*≥*a*. *integrable* ($\lambda u.$ g' *u* / *u powr* (*p* + *1*)) *a b*
and g'*-bounded*: *eventually* (λa::*real*. (∀ *b*>*a*. ∃ *c*. ∀ *x*∈{*a*..*b*}. g' *x* ≤ *c*)) *at-top*
and *g-bigomega*: *g* ∈ Ω($\lambda x.$ g' (*real x*))
and g'*-nonneg*: *eventually* ($\lambda x.$ g' *x* ≥ *0*) *at-top*
begin

definition *gc2* ≡ *SOME gc2*. *gc2* > *0* ∧ *eventually* ($\lambda x.$ *g x* ≥ *gc2* ∗ g' (*real x*)) *at-top*

lemma *gc2*: *gc2* > *0 eventually* ($\lambda x.$ *g x* ≥ *gc2* ∗ g' (*real x*)) *at-top*
⟨*proof*⟩

definition *gx0* ≡ *max* x_1 (*SOME gx0*. ∀ *x*≥*gx0*. *g x* ≥ *gc2* ∗ g' (*real x*) ∧ *f x* > *0* ∧ g' (*real x*) ≥ *0*)
definition *gx1* ≡ *max gx0* (*SOME gx1*. ∀ *x*≥*gx1*. ∀ *i*<*k*. (*ts*!*i*) *x* ≥ *gx0*)

lemma *gx0*:
assumes *x* ≥ *gx0*
shows *g x* ≥ *gc2* ∗ g' (*real x*) *f x* > *0* g' (*real x*) ≥ *0*
⟨*proof*⟩

lemma *gx1*:
assumes *x* ≥ *gx1 i* < *k*
shows (*ts*!*i*) *x* ≥ *gx0*
⟨*proof*⟩

lemma *gx0-ge-x1*: *gx0* ≥ x_1 ⟨*proof*⟩
lemma *gx0-le-gx1*: *gx0* ≤ *gx1* ⟨*proof*⟩

function $f2'$ :: *nat* ⇒ *real* **where**
x < *gx1* ⟹ $f2'$ *x* = *max 0* (*f x* / *gc2*)
| *x* ≥ *gx1* ⟹ $f2'$ *x* = g' (*real x*) + ($\sum$ *i*<*k*. *as*!*i*∗$f2'$ ((*ts*!*i*) *x*))
⟨*proof*⟩
termination ⟨*proof*⟩

lemma $f2'$*-nonneg*: *x* ≥ *gx0* ⟹ $f2'$ *x* ≥ *0*
⟨*proof*⟩

lemma $f2'$*-le-f*: *x* ≥ x_0 ⟹ *gc2* ∗ $f2'$ *x* ≤ *f x*
⟨*proof*⟩

lemma $f2'$*-pos*: *eventually* ($\lambda x.$ $f2'$ *x* > *0*) *at-top*
⟨*proof*⟩

lemma *bigomega-f-aux*:
  obtains a **where** $a \geq A$ $\forall a' \geq a.\ a' \in \mathbb{N} \longrightarrow$
    $f \in \Omega(\lambda x.\ x\ powr\ p * (1 + integral\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a'\ x))$
⟨*proof*⟩

lemma *bigomega-f*:
  obtains a **where** $a \geq A$ $f \in \Omega(\lambda x.\ x\ powr\ p * (1 + integral\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p+1))\ a\ x))$
⟨*proof*⟩

end

locale *akra-bazzi-upper* = *akra-bazzi-function* +
  fixes $g' :: real \Rightarrow real$
  assumes *g′-integrable*: $\exists a.\ \forall b \geq a.\ integrable\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a\ b$
  and *g-growth1*: $\exists C\ c1.\ c1 > 0 \wedge C < Min\ (set\ bs) \wedge$
    $eventually\ (\lambda x.\ \forall u \in \{C*x..x\}.\ g'\ u \geq c1 * g'\ x)\ at\text{-}top$
  and *g-bigo*: $g \in O(g')$
  and *g′-nonneg*: $eventually\ (\lambda x.\ g'\ x \geq 0)\ at\text{-}top$
begin

definition $gc1 \equiv SOME\ gc1.\ gc1 > 0 \wedge eventually\ (\lambda x.\ g\ x \leq gc1 * g'\ (real\ x))\ at\text{-}top$

lemma *gc1*: $gc1 > 0$ $eventually\ (\lambda x.\ g\ x \leq gc1 * g'\ (real\ x))\ at\text{-}top$
⟨*proof*⟩

definition $gx3 \equiv max\ x_1\ (SOME\ gx0.\ \forall x \geq gx0.\ g\ x \leq gc1 * g'\ (real\ x))$

lemma *gx3*:
  assumes $x \geq gx3$
  shows $g\ x \leq gc1 * g'\ (real\ x)$
⟨*proof*⟩

lemma *gx3-ge-x1*: $gx3 \geq x_1$ ⟨*proof*⟩

function $f' :: nat \Rightarrow real$ **where**
  $x < gx3 \Longrightarrow f'\ x = max\ 0\ (f\ x\ /\ gc1)$
| $x \geq gx3 \Longrightarrow f'\ x = g'\ (real\ x) + (\sum i<k.\ as!i * f'\ ((ts!i)\ x))$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *f′-ge-f*: $x \geq x_0 \Longrightarrow gc1 * f'\ x \geq f\ x$
⟨*proof*⟩

lemma *bigo-f-aux*:
  obtains *a* **where** $a \geq A$ $\forall a' \geq a.\ a' \in \mathbb{N} \longrightarrow$
    $f \in O(\lambda x.\ x\ powr\ p * (1 + integral\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a'\ x))$
⟨*proof*⟩

lemma *bigo-f*:
  obtains *a* **where** $a > A$ $f \in O(\lambda x.\ x\ powr\ p * (1 + integral\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a\ x))$
⟨*proof*⟩

end

locale *akra-bazzi* = *akra-bazzi-function* +
  fixes $g' :: real \Rightarrow real$
  assumes *f-pos*: *eventually* $(\lambda x.\ f\ x > 0)$ *at-top*
  and *g'-nonneg*: *eventually* $(\lambda x.\ g'\ x \geq 0)$ *at-top*
  assumes *g'-integrable*: $\exists a.\ \forall b \geq a.\ integrable\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a\ b$
  and *g-growth1*: $\exists C\ c1.\ c1 > 0 \wedge C < Min\ (set\ bs) \wedge$
    eventually $(\lambda x.\ \forall u \in \{C*x..x\}.\ g'\ u \geq c1 * g'\ x)$ *at-top*
  and *g-growth2*: $\exists C\ c2.\ c2 > 0 \wedge C < Min\ (set\ bs) \wedge$
    eventually $(\lambda x.\ \forall u \in \{C*x..x\}.\ g'\ u \leq c2 * g'\ x)$ *at-top*
  and *g-bounded*: *eventually* $(\lambda a::real.\ (\forall b>a.\ \exists c.\ \forall x \in \{a..b\}.\ g'\ x \leq c))$ *at-top*
  and *g-bigtheta*: $g \in \Theta(g')$
begin

sublocale *akra-bazzi-lower* ⟨*proof*⟩
sublocale *akra-bazzi-upper* ⟨*proof*⟩

lemma *bigtheta-f*:
  obtains *a* **where** $a > A$ $f \in \Theta(\lambda x.\ x\ powr\ p * (1 + integral\ (\lambda u.\ g'\ u\ /\ u\ powr\ (p + 1))\ a\ x))$
⟨*proof*⟩

end

named-theorems *akra-bazzi-term-intros introduction rules for Akra−Bazzi terms*

lemma *akra-bazzi-term-floor-add* [*akra-bazzi-term-intros*]:
  assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 + c$ $c < (1 - b) * real\ x_1$ $x_1 > 0$
  shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lfloor b*real\ x + c \rfloor)$
⟨*proof*⟩

lemma *akra-bazzi-term-floor-add'* [*akra-bazzi-term-intros*]:
  assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 + real\ c$ $real\ c < (1 - b) * real\ x_1$ $x_1 > 0$
  shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lfloor b*real\ x \rfloor + c)$

⟨*proof*⟩

lemma *akra-bazzi-term-floor-subtract* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 - c$ $0 < c + (1 - b) * real\ x_1$ $x_1 > 0$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lfloor b*real\ x - c\rfloor)$
⟨*proof*⟩

lemma *akra-bazzi-term-floor-subtract′* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 - real\ c$ $0 < real\ c + (1 - b) * real\ x_1$ $x_1 > 0$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lfloor b*real\ x\rfloor - c)$
⟨*proof*⟩

lemma *akra-bazzi-term-floor* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1$ $0 < (1 - b) * real\ x_1$ $x_1 > 0$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lfloor b*real\ x\rfloor)$
⟨*proof*⟩

lemma *akra-bazzi-term-ceiling-add* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 + c$ $c + 1 \leq (1 - b) * x_1$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lceil b*real\ x + c\rceil)$
⟨*proof*⟩

lemma *akra-bazzi-term-ceiling-add′* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 + real\ c$ $real\ c + 1 \leq (1 - b) * x_1$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lceil b*real\ x\rceil + c)$
⟨*proof*⟩

lemma *akra-bazzi-term-ceiling-subtract* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 - c$ $1 \leq c + (1 - b) * x_1$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lceil b*real\ x - c\rceil)$
⟨*proof*⟩

lemma *akra-bazzi-term-ceiling-subtract′* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1 - real\ c$ $1 \leq real\ c + (1 - b) * x_1$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lceil b*real\ x\rceil - c)$
⟨*proof*⟩

lemma *akra-bazzi-term-ceiling* [*akra-bazzi-term-intros*]:
assumes $(b::real) > 0$ $b < 1$ $real\ x_0 \leq b * real\ x_1$ $1 \leq (1 - b) * x_1$
shows $akra\text{-}bazzi\text{-}term\ x_0\ x_1\ b\ (\lambda x.\ nat\ \lceil b*real\ x\rceil)$
⟨*proof*⟩

end

5 The Master theorem

theory *Master-Theorem*
imports
HOL−Analysis.Equivalence-Lebesgue-Henstock-Integration
Akra-Bazzi-Library
Akra-Bazzi
begin

lemma *fundamental-theorem-of-calculus-real*:
$a \leq b \Longrightarrow \forall x{\in}\{a..b\}.\ (f\ \textit{has-real-derivative}\ f'\ x)\ (\textit{at}\ x\ \textit{within}\ \{a..b\}) \Longrightarrow$
$(f'\ \textit{has-integral}\ (f\ b - f\ a))\ \{a..b\}$
⟨*proof*⟩

lemma *integral-powr*:
$y \neq -1 \Longrightarrow a \leq b \Longrightarrow a > 0 \Longrightarrow \textit{integral}\ \{a..b\}\ (\lambda x.\ x\ \textit{powr}\ y :: \textit{real}) =$
$\textit{inverse}\ (y + 1) * (b\ \textit{powr}\ (y + 1) - a\ \textit{powr}\ (y + 1))$
⟨*proof*⟩

lemma *integral-ln-powr-over-x*:
$y \neq -1 \Longrightarrow a \leq b \Longrightarrow a > 1 \Longrightarrow \textit{integral}\ \{a..b\}\ (\lambda x.\ \textit{ln}\ x\ \textit{powr}\ y\ /\ x :: \textit{real}) =$
$\textit{inverse}\ (y + 1) * (\textit{ln}\ b\ \textit{powr}\ (y + 1) - \textit{ln}\ a\ \textit{powr}\ (y + 1))$
⟨*proof*⟩

lemma *integral-one-over-x-ln-x*:
$a \leq b \Longrightarrow a > 1 \Longrightarrow \textit{integral}\ \{a..b\}\ (\lambda x.\ \textit{inverse}\ (x * \textit{ln}\ x) :: \textit{real}) = \textit{ln}\ (\textit{ln}\ b)$
$- \textit{ln}\ (\textit{ln}\ a)$
⟨*proof*⟩

lemma *akra-bazzi-integral-kurzweil-henstock*:
akra-bazzi-integral $(\lambda f\ a\ b.\ f\ \textit{integrable-on}\ \{a..b\})\ (\lambda f\ a\ b.\ \textit{integral}\ \{a..b\}\ f)$
⟨*proof*⟩

locale *master-theorem-function* = *akra-bazzi-recursion* +
fixes $g :: \textit{nat} \Rightarrow \textit{real}$
assumes *f-nonneg-base*: $x \geq x_0 \Longrightarrow x < x_1 \Longrightarrow f\ x \geq 0$
and *f-rec*: $x \geq x_1 \Longrightarrow f\ x = g\ x + (\sum i{<}k.\ \textit{as}!i * f\ ((\textit{ts}!i)\ x))$
and *g-nonneg*: $x \geq x_1 \Longrightarrow g\ x \geq 0$
and *ex-pos-a*: $\exists\, a{\in}\textit{set as}.\ a > 0$
begin

interpretation *akra-bazzi-integral* $\lambda f\ a\ b.\ f\ \textit{integrable-on}\ \{a..b\}\ \lambda f\ a\ b.\ \textit{integral}$
$\{a..b\}\ f$
⟨*proof*⟩

sublocale *akra-bazzi-function* x_0 x_1 *k as bs ts f* λf *a b. f integrable-on* $\{a..b\}$
λf *a b. integral* $\{a..b\}$ *f g*
⟨*proof*⟩

context
begin

private lemma *g-nonneg′*: *eventually* ($\lambda x.$ $g\ x \geq 0$) *at-top*
⟨*proof*⟩ **lemma** *g-pos*:
assumes $g \in \Omega(h)$
assumes *eventually* ($\lambda x.$ $h\ x > 0$) *at-top*
shows *eventually* ($\lambda x.$ $g\ x > 0$) *at-top*
⟨*proof*⟩ **lemma** *f-pos*:
assumes $g \in \Omega(h)$
assumes *eventually* ($\lambda x.$ $h\ x > 0$) *at-top*
shows *eventually* ($\lambda x.$ $f\ x > 0$) *at-top*
⟨*proof*⟩

lemma *bs-lower-bound*: $\exists\, C>0.\ \forall\, b\in set\ bs.\ C < b$
⟨*proof*⟩ **lemma** *powr-growth2*:
$\exists\, C\ c2.\ 0 < c2 \wedge C < Min\ (set\ bs) \wedge$
eventually ($\lambda x.\ \forall\, u\in\{C * x..x\}.\ c2 * x\ powr\ p' \geq u\ powr\ p'$) *at-top*
⟨*proof*⟩ **lemma** *powr-growth1*:
$\exists\, C\ c1.\ 0 < c1 \wedge C < Min\ (set\ bs) \wedge$
eventually ($\lambda x.\ \forall\, u\in\{C * x..x\}.\ c1 * x\ powr\ p' \leq u\ powr\ p'$) *at-top*
⟨*proof*⟩ **lemma** *powr-ln-powr-lower-bound*:
$a > 1 \Longrightarrow a \leq x \Longrightarrow x \leq b \Longrightarrow$
$min\ (a\ powr\ p)\ (b\ powr\ p) * min\ (ln\ a\ powr\ p')\ (ln\ b\ powr\ p') \leq x\ powr\ p * ln\ x\ powr\ p'$
⟨*proof*⟩ **lemma** *powr-ln-powr-upper-bound*:
$a > 1 \Longrightarrow a \leq x \Longrightarrow x \leq b \Longrightarrow$
$max\ (a\ powr\ p)\ (b\ powr\ p) * max\ (ln\ a\ powr\ p')\ (ln\ b\ powr\ p') \geq x\ powr\ p * ln\ x\ powr\ p'$
⟨*proof*⟩ **lemma** *powr-ln-powr-upper-bound′*:
eventually ($\lambda a.\ \forall\, b>a.\ \exists\, c.\ \forall\, x\in\{a..b\}.\ x\ powr\ p * ln\ x\ powr\ p' \leq c$) *at-top*
⟨*proof*⟩ **lemma** *powr-upper-bound′*:
eventually ($\lambda a::real.\ \forall\, b>a.\ \exists\, c.\ \forall\, x\in\{a..b\}.\ x\ powr\ p' \leq c$) *at-top*
⟨*proof*⟩

lemmas *bounds* =
powr-ln-powr-lower-bound powr-ln-powr-upper-bound powr-ln-powr-upper-bound′ powr-upper-bound′

private lemma *eventually-ln-const*:
assumes $(C::real) > 0$
shows *eventually* ($\lambda x.\ ln\ (C*x)\ /\ ln\ x > 1/2$) *at-top*
⟨*proof*⟩ **lemma** *powr-ln-powr-growth1*: $\exists\, C\ c1.\ 0 < c1 \wedge C < Min\ (set\ bs) \wedge$
eventually ($\lambda x.\ \forall\, u\in\{C * x..x\}.\ c1 * (x\ powr\ r * ln\ x\ powr\ r') \leq u\ powr\ r * ln$

u powr r') *at-top*
⟨*proof*⟩ **lemma** *powr-ln-powr-growth2*: ∃ *C c1*. *0* < *c1* ∧ *C* < *Min* (*set bs*) ∧
eventually (λ*x*. ∀ *u*∈{*C* ∗ *x*..*x*}. *c1* ∗ (*x powr r* ∗ *ln x powr r'*) ≥ *u powr r* ∗ *ln*
u powr r') *at-top*
⟨*proof*⟩

lemmas *growths* = *powr-growth1 powr-growth2 powr-ln-powr-growth1 powr-ln-powr-growth2*

private lemma *master-integrable*:
∃ *a*::*real*. ∀ *b*≥*a*. (λ*u*. *u powr r* ∗ *ln u powr s* / *u powr t*) *integrable-on* {*a*..*b*}
∃ *a*::*real*. ∀ *b*≥*a*. (λ*u*. *u powr r* / *u powr s*) *integrable-on* {*a*..*b*}
⟨*proof*⟩ **lemma** *master-integral*:
fixes *a p p'* :: *real*
assumes *p*: *p* ≠ *p'* **and** *a*: *a* > *0*
obtains *c d* **where** *c* ≠ *0* *p* > *p'* ⟶ *d* ≠ *0*
(λ*x*::*nat*. *x powr p* ∗ (*1* + *integral* {*a*..*x*} (λ*u*. *u powr p'* / *u powr* (*p*+*1*)))) ∈
Θ(λ*x*::*nat*. *d* ∗ *x powr p* + *c* ∗ *x powr p'*)
⟨*proof*⟩ **lemma** *master-integral'*:
fixes *a p p'* :: *real*
assumes *p'*: *p'* ≠ *0* **and** *a*: *a* > *1*
obtains *c d* :: *real* **where** *p'* < *0* ⟶ *c* ≠ *0* *d* ≠ *0*
(λ*x*::*nat*. *x powr p* ∗ (*1* + *integral* {*a*..*x*} (λ*u*. *u powr p* ∗ *ln u powr* (*p'*−*1*) / *u*
powr (*p*+*1*)))) ∈
Θ(λ*x*::*nat*. *c* ∗ *x powr p* + *d* ∗ *x powr p* ∗ *ln x powr p'*)
⟨*proof*⟩ **lemma** *master-integral''*:
fixes *a p p'* :: *real*
assumes *a*: *a* > *1*
shows (λ*x*::*nat*. *x powr p* ∗ (*1* + *integral* {*a*..*x*} (λ*u*. *u powr p* ∗ *ln u powr* −
1/*u powr* (*p*+*1*)))) ∈
Θ(λ*x*::*nat*. *x powr p* ∗ *ln* (*ln x*))
⟨*proof*⟩

lemma *master1-bigo*:
assumes *g-bigo*: *g* ∈ *O*(λ*x*. *real x powr p'*)
assumes *less-p'*: (∑ *i*<*k*. *as*!*i* ∗ *bs*!*i powr p'*) > *1*
shows *f* ∈ *O*(λ*x*. *real x powr p*)
⟨*proof*⟩

lemma *master1*:
assumes *g-bigo*: *g* ∈ *O*(λ*x*. *real x powr p'*)
assumes *less-p'*: (∑ *i*<*k*. *as*!*i* ∗ *bs*!*i powr p'*) > *1*
assumes *f-pos*: *eventually* (λ*x*. *f x* > *0*) *at-top*
shows *f* ∈ Θ(λ*x*. *real x powr p*)
⟨*proof*⟩

lemma *master2-3*:
  assumes *g-bigtheta*: $g \in \Theta(\lambda x.\ real\ x\ powr\ p * ln\ (real\ x)\ powr\ (p' - 1))$
  assumes *p'*: $p' > 0$
  shows $f \in \Theta(\lambda x.\ real\ x\ powr\ p * ln\ (real\ x)\ powr\ p')$
⟨*proof*⟩

lemma *master2-1*:
  assumes *g-bigtheta*: $g \in \Theta(\lambda x.\ real\ x\ powr\ p * ln\ (real\ x)\ powr\ p')$
  assumes *p'*: $p' < -1$
  shows $f \in \Theta(\lambda x.\ real\ x\ powr\ p)$
⟨*proof*⟩

lemma *master2-2*:
  assumes *g-bigtheta*: $g \in \Theta(\lambda x.\ real\ x\ powr\ p\ /\ ln\ (real\ x))$
  shows $f \in \Theta(\lambda x.\ real\ x\ powr\ p * ln\ (ln\ (real\ x)))$
⟨*proof*⟩

lemma *master3*:
  assumes *g-bigtheta*: $g \in \Theta(\lambda x.\ real\ x\ powr\ p')$
  assumes *p'-greater'*: $(\sum i{<}k.\ as!i * bs!i\ powr\ p') < 1$
  shows $f \in \Theta(\lambda x.\ real\ x\ powr\ p')$
⟨*proof*⟩

end
end

end

6 Evaluating expressions with rational numerals

theory *Eval-Numeral*
imports
  Complex-Main
begin

lemma *real-numeral-to-Ratreal*:
  $(0::real) = Ratreal\ (Frct\ (0,\ 1))$
  $(1::real) = Ratreal\ (Frct\ (1,\ 1))$
  $(numeral\ x :: real) = Ratreal\ (Frct\ (numeral\ x,\ 1))$
  $(1::int) = numeral\ Num.One$
  ⟨*proof*⟩

lemma *real-equals-code*: $Ratreal\ x = Ratreal\ y \longleftrightarrow x = y$
  ⟨*proof*⟩

lemma *Rat-normalize-idempotent*: $Rat.normalize\ (Rat.normalize\ x) = Rat.normalize\ x$
⟨*proof*⟩

lemma *uminus-pow-Numeral1*: $(-(x::-::monoid\text{-}mult))$ ^ *Numeral1* $= -x$ ⟨*proof*⟩

lemmas *power-numeral-simps* = *power-0 uminus-pow-Numeral1 power-minus-Bit0 power-minus-Bit1*

lemma *Fract-normalize*: *Fract* (*fst* (*Rat.normalize* (*x*,*y*))) (*snd* (*Rat.normalize* (*x*,*y*))) = *Fract x y*
⟨*proof*⟩

lemma *Frct-add*: *Frct* (*a*, *numeral b*) + *Frct* (*c*, *numeral d*) =
Frct (*Rat.normalize* (*a* ∗ *numeral d* + *c* ∗ *numeral b*, *numeral* (*b*∗*d*)))
⟨*proof*⟩

lemma *Frct-uminus*: −(*Frct* (*a*,*b*)) = *Frct* (−*a*,*b*) ⟨*proof*⟩

lemma *Frct-diff*: *Frct* (*a*, *numeral b*) − *Frct* (*c*, *numeral d*) =
Frct (*Rat.normalize* (*a* ∗ *numeral d* − *c* ∗ *numeral b*, *numeral* (*b*∗*d*)))
⟨*proof*⟩

lemma *Frct-mult*: *Frct* (*a*, *numeral b*) ∗ *Frct* (*c*, *numeral d*) = *Frct* (*a*∗*c*, *numeral* (*b*∗*d*))
⟨*proof*⟩

lemma *Frct-inverse*: *inverse* (*Frct* (*a*, *b*)) = *Frct* (*b*, *a*) ⟨*proof*⟩

lemma *Frct-divide*: *Frct* (*a*, *numeral b*) / *Frct* (*c*, *numeral d*) = *Frct* (*a*∗*numeral d*, *numeral b* ∗ *c*)
⟨*proof*⟩

lemma *Frct-pow*: *Frct* (*a*, *numeral b*) ^ *c* = *Frct* (*a* ^ *c*, *numeral b* ^ *c*)
⟨*proof*⟩

lemma *Frct-less*: *Frct* (*a*, *numeral b*) < *Frct* (*c*, *numeral d*) ⟷ *a* ∗ *numeral d* < *c* ∗ *numeral b*
⟨*proof*⟩

lemma *Frct-le*: *Frct* (*a*, *numeral b*) ≤ *Frct* (*c*, *numeral d*) ⟷ *a* ∗ *numeral d* ≤ *c* ∗ *numeral b*
⟨*proof*⟩

lemma *Frct-equals*: *Frct* (*a*, *numeral b*) = *Frct* (*c*, *numeral d*) ⟷ *a* ∗ *numeral d* = *c* ∗ *numeral b*
⟨*proof*⟩

lemma *real-power-code*: (*Ratreal x*) ^ *y* = *Ratreal* (*x* ^ *y*) ⟨*proof*⟩

lemmas *real-arith-code* =
real-plus-code real-minus-code real-times-code real-uminus-code real-inverse-code real-divide-code real-power-code real-less-code real-less-eq-code real-equals-code

lemmas *rat-arith-code* =
Frct-add Frct-uminus Frct-diff Frct-mult Frct-inverse Frct-divide Frct-pow Frct-less Frct-le Frct-equals

lemma *gcd-numeral-red*: *gcd* (*numeral x*::*int*) (*numeral y*) = *gcd* (*numeral y*) (*numeral x mod numeral y*)
⟨*proof*⟩

lemma *divmod-one*:
divmod (*Num.One*) (*Num.One*) = (*Numeral1*, *0*)
divmod (*Num.One*) (*Num.Bit0 x*) = (*0*, *Numeral1*)
divmod (*Num.One*) (*Num.Bit1 x*) = (*0*, *Numeral1*)
divmod x (*Num.One*) = (*numeral x*, *0*)
⟨*proof*⟩

lemmas *divmod-numeral-simps* =
div-0 div-by-0 mod-0 mod-by-0
fst-divmod [*symmetric*]
snd-divmod [*symmetric*]
divmod-cancel
divmod-steps [*simplified rel-simps if-True*] *divmod-trivial*
rel-simps

lemma *Suc-0-to-numeral*: *Suc 0* = *Numeral1* ⟨*proof*⟩
lemmas *Suc-to-numeral* = *Suc-0-to-numeral Num.Suc-1 Num.Suc-numeral*

lemma *rat-powr*:
0 powr y = *0*
x > *0* ⟹ *x powr Ratreal* (*Frct* (*0*, *Numeral1*)) = *Ratreal* (*Frct* (*Numeral1*, *Numeral1*))
x > *0* ⟹ *x powr Ratreal* (*Frct* (*numeral a*, *Numeral1*)) = *x* ^ *numeral a*
x > *0* ⟹ *x powr Ratreal* (*Frct* (−*numeral a*, *Numeral1*)) = *inverse* (*x* ^ *numeral a*)
⟨*proof*⟩

lemmas *eval-numeral-simps* =
real-numeral-to-Ratreal real-arith-code rat-arith-code Num.arith-simps Rat.normalize-def fst-conv snd-conv gcd-0-int gcd-0-left-int gcd.bottom-right-bottom gcd.bottom-left-bottom
gcd-neg1-int gcd-neg2-int gcd-numeral-red zmod-numeral-Bit0 zmod-numeral-Bit1 power-numeral-simps
divmod-numeral-simps numeral-One [*symmetric*] *Groups.Let-0 Num.Let-numeral Suc-to-numeral power-numeral*
greaterThanLessThan-iff atLeastAtMost-iff atLeastLessThan-iff greaterThanAtMost-iff rat-powr

Num.pow.simps Num.sqr.simps Product-Type.split of-int-numeral of-int-neg-numeral of-nat-numeral

⟨*ML*⟩

lemma *2125438754865958951 2∗3142135236324643574538843 61∗2342523623324234∗56432743858724173474*
12561712738645824362329316482973164398214286 powr 2 /
(113024631297842312 3+231212374631082764842731842∗122474378389424362347451251263)
>
(123132445129312472435432797686457459294758293106512056238 44::real)
⟨*proof*⟩

end

7 The proof methods

7.1 Master theorem and termination

theory *Akra-Bazzi-Method*
imports
Complex-Main
Akra-Bazzi
Master-Theorem
Eval-Numeral
begin

lemma *landau-symbol-ge-3-cong*:
assumes *landau-symbol L L′ Lr*
assumes $\bigwedge x::'a::linordered\text{-}semidom.\ x \geq 3 \Longrightarrow f\ x = g\ x$
shows *L at-top (f) = L at-top (g)*
⟨*proof*⟩

lemma *exp-1-lt-3*: *exp (1::real) < 3*
⟨*proof*⟩

lemma *ln-ln-pos*:
assumes *(x::real) ≥ 3*
shows *ln (ln x) > 0*
⟨*proof*⟩

definition *akra-bazzi-terms* **where**
akra-bazzi-terms x_0 x_1 *bs ts* = (∀ *i<length bs. akra-bazzi-term* x_0 x_1 *(bs!i) (ts!i)*)

lemma *akra-bazzi-termsI*:
(⋀*i. i < length bs* ⟹ *akra-bazzi-term* x_0 x_1 *(bs!i) (ts!i)*) ⟹ *akra-bazzi-terms* x_0 x_1 *bs ts*
⟨*proof*⟩

lemma *master-theorem-functionI*:

assumes $\forall x \in \{x_0..<x_1\}.\ f\ x \geq 0$
assumes $\forall x \geq x_1.\ f\ x = g\ x + (\sum i<k.\ as\ !\ i * f\ ((ts\ !\ i)\ x))$
assumes $\forall x \geq x_1.\ g\ x \geq 0$
assumes $\forall a \in set\ as.\ a \geq 0$
assumes *list-ex* $(\lambda a.\ a > 0)$ *as*
assumes $\forall b \in set\ bs.\ b \in \{0<..<1\}$
assumes $k \neq 0$
assumes *length as* = *k*
assumes *length bs* = *k*
assumes *length ts* = *k*
assumes *akra-bazzi-terms* x_0 x_1 *bs ts*
shows *master-theorem-function* x_0 x_1 *k as bs ts f g*
⟨*proof*⟩

lemma *akra-bazzi-term-measure*:
$x \geq x_1 \Longrightarrow$ *akra-bazzi-term 0* x_1 *b t* $\Longrightarrow (t\ x,\ x) \in$ *Wellfounded.measure* $(\lambda n::nat.\ n)$
$x > x_1 \Longrightarrow$ *akra-bazzi-term 0 (Suc* x_1*) b t* $\Longrightarrow (t\ x,\ x) \in$ *Wellfounded.measure* $(\lambda n::nat.\ n)$
⟨*proof*⟩

lemma *measure-prod-conv*:
$((a,\ b),\ (c,\ d)) \in$ *Wellfounded.measure* $(\lambda x.\ t\ (fst\ x)) \longleftrightarrow (a,\ c) \in$ *Wellfounded.measure t*
$((e,\ f),\ (g,\ h)) \in$ *Wellfounded.measure* $(\lambda x.\ t\ (snd\ x)) \longleftrightarrow (f,\ h) \in$ *Wellfounded.measure t*
⟨*proof*⟩

lemmas *measure-prod-conv'* = *measure-prod-conv*[**where** $t = \lambda x.\ x$]

lemma *akra-bazzi-termination-simps*:
fixes *x* :: *nat*
shows $a * real\ x\ /\ b = a/b * real\ x$ $real\ x\ /\ b = 1/b * real\ x$
⟨*proof*⟩

lemma *akra-bazzi-params-nonzeroI*:
length as = *length bs* $\Longrightarrow$
$(\forall a \in set\ as.\ a \geq 0) \Longrightarrow (\forall b \in set\ bs.\ b \in \{0<..<1\}) \Longrightarrow (\exists a \in set\ as.\ a > 0) \Longrightarrow$
akra-bazzi-params-nonzero (length as) as bs ⟨*proof*⟩

lemmas *akra-bazzi-p-rel-intros* =
akra-bazzi-params-nonzero.p-lessI[*rotated*, *OF - akra-bazzi-params-nonzeroI*]
akra-bazzi-params-nonzero.p-greaterI[*rotated*, *OF - akra-bazzi-params-nonzeroI*]
akra-bazzi-params-nonzero.p-leI[*rotated*, *OF - akra-bazzi-params-nonzeroI*]
akra-bazzi-params-nonzero.p-geI[*rotated*, *OF - akra-bazzi-params-nonzeroI*]
akra-bazzi-params-nonzero.p-boundsI[*rotated*, *OF - akra-bazzi-params-nonzeroI*]
akra-bazzi-params-nonzero.p-boundsI'[*rotated*, *OF - akra-bazzi-params-nonzeroI*]

lemma *eval-length*: *length* [] = *0* *length* (*x* # *xs*) = *Suc* (*length xs*) ⟨*proof*⟩

lemma *eval-akra-bazzi-sum*:
$(\sum i<0.\ as!i * bs!i\ powr\ x) = 0$
$(\sum i<Suc\ 0.\ (a\#as)!i * (b\#bs)!i\ powr\ x) = a * b\ powr\ x$
$(\sum i<Suc\ k.\ (a\#as)!i * (b\#bs)!i\ powr\ x) = a * b\ powr\ x + (\sum i<k.\ as!i * bs!i\ powr\ x)$
⟨*proof*⟩

lemma *eval-akra-bazzi-sum'*:
$(\sum i<0.\ as!i * f\ ((ts!i)\ x)) = 0$
$(\sum i<Suc\ 0.\ (a\#as)!i * f\ (((t\#ts)!i)\ x)) = a * f\ (t\ x)$
$(\sum i<Suc\ k.\ (a\#as)!i * f\ (((t\#ts)!i)\ x)) = a * f\ (t\ x) + (\sum i<k.\ as!i * f\ ((ts!i)\ x))$
⟨*proof*⟩

lemma *akra-bazzi-termsI'*:
akra-bazzi-terms x_0 x_1 [] []
akra-bazzi-term x_0 x_1 *b t* $\Longrightarrow$ *akra-bazzi-terms* x_0 x_1 *bs ts* $\Longrightarrow$ *akra-bazzi-terms* x_0 x_1 $(b\#bs)$ $(t\#ts)$
⟨*proof*⟩

lemma *ball-set-intros*: $(\forall x\in set\ [].\ P\ x)$ $P\ x \Longrightarrow (\forall x\in set\ xs.\ P\ x) \Longrightarrow (\forall x\in set\ (x\#xs).\ P\ x)$
⟨*proof*⟩

lemma *ball-set-simps*: $(\forall x\in set\ [].\ P\ x) = True$ $(\forall x\in set\ (x\#xs).\ P\ x) = (P\ x \wedge (\forall x\in set\ xs.\ P\ x))$
⟨*proof*⟩

lemma *bex-set-simps*: $(\exists x\in set\ [].\ P\ x) = False$ $(\exists x\in set\ (x\#xs).\ P\ x) = (P\ x \vee (\exists x\in set\ xs.\ P\ x))$
⟨*proof*⟩

lemma *eval-akra-bazzi-le-list-ex*:
list-ex P $(x\#y\#xs) \longleftrightarrow P\ x \vee$ *list-ex* P $(y\#xs)$
list-ex P $[x] \longleftrightarrow P\ x$
list-ex P $[] \longleftrightarrow False$
⟨*proof*⟩

lemma *eval-akra-bazzi-le-sum-list*:
$x \le$ *sum-list* $[] \longleftrightarrow x \le 0$ $x \le$ *sum-list* $(y\#ys) \longleftrightarrow x \le y +$ *sum-list* *ys*
$x \le z +$ *sum-list* $[] \longleftrightarrow x \le z$ $x \le z +$ *sum-list* $(y\#ys) \longleftrightarrow x \le z + y +$ *sum-list* *ys*
⟨*proof*⟩

lemma *atLeastLessThanE*: $x \in \{a..<b\} \Longrightarrow (x \ge a \Longrightarrow x < b \Longrightarrow P) \Longrightarrow P$
⟨*proof*⟩

lemma *master-theorem-preprocess*:

$\Theta(\lambda n{::}nat.\ 1) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 0)$
$\Theta(\lambda n{::}nat.\ real\ n) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 1)$
$O(\lambda n{::}nat.\ 1) = O(\lambda n{::}nat.\ real\ n\ powr\ 0)$
$O(\lambda n{::}nat.\ real\ n) = O(\lambda n{::}nat.\ real\ n\ powr\ 1)$

$\Theta(\lambda n{::}nat.\ ln\ (ln\ (real\ n))) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 0 * ln\ (ln\ (real\ n)))$
$\Theta(\lambda n{::}nat.\ real\ n * ln\ (ln\ (real\ n))) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 1 * ln\ (ln\ (real\ n)))$

$\Theta(\lambda n{::}nat.\ ln\ (real\ n)) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 0 * ln\ (real\ n)\ powr\ 1)$
$\Theta(\lambda n{::}nat.\ real\ n * ln\ (real\ n)) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 1 * ln\ (real\ n)\ powr\ 1)$
$\Theta(\lambda n{::}nat.\ real\ n\ powr\ p * ln\ (real\ n)) = \Theta(\lambda n{::}nat.\ real\ n\ powr\ p * ln\ (real\ n)\ powr\ 1)$
$\Theta(\lambda n{::}nat.\ ln\ (real\ n)\ powr\ p') = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 0 * ln\ (real\ n)\ powr\ p')$
$\Theta(\lambda n{::}nat.\ real\ n * ln\ (real\ n)\ powr\ p') = \Theta(\lambda n{::}nat.\ real\ n\ powr\ 1 * ln\ (real\ n)\ powr\ p')$
$\langle proof \rangle$

lemma *akra-bazzi-term-imp-size-less*:
$x_1 \le x \Longrightarrow akra\text{-}bazzi\text{-}term\ 0\ x_1\ b\ t \Longrightarrow size\ (t\ x) < size\ x$
$x_1 < x \Longrightarrow akra\text{-}bazzi\text{-}term\ 0\ (Suc\ x_1)\ b\ t \Longrightarrow size\ (t\ x) < size\ x$
$\langle proof \rangle$

definition $CLAMP\ (f :: nat \Rightarrow real)\ x = (if\ x < 3\ then\ 0\ else\ f\ x)$
definition $CLAMP'\ (f :: nat \Rightarrow real)\ x = (if\ x < 3\ then\ 0\ else\ f\ x)$
definition $MASTER\text{-}BOUND\ a\ b\ c\ x = real\ x\ powr\ a * ln\ (real\ x)\ powr\ b * ln\ (ln\ (real\ x))\ powr\ c$
definition $MASTER\text{-}BOUND'\ a\ b\ x = real\ x\ powr\ a * ln\ (real\ x)\ powr\ b$
definition $MASTER\text{-}BOUND''\ a\ x = real\ x\ powr\ a$

lemma *ln-1-imp-less-3*:
$ln\ x = (1{::}real) \Longrightarrow x < 3$
$\langle proof \rangle$

lemma *ln-1-imp-less-3'*: $ln\ (real\ (x{::}nat)) = 1 \Longrightarrow x < 3\ \langle proof \rangle$

lemma *ln-ln-nonneg*: $x \ge (3{::}real) \Longrightarrow ln\ (ln\ x) \ge 0\ \langle proof \rangle$
lemma *ln-ln-nonneg'*: $x \ge (3{::}nat) \Longrightarrow ln\ (ln\ (real\ x)) \ge 0\ \langle proof \rangle$

lemma *MASTER-BOUND-postproc*:
$CLAMP\ (MASTER\text{-}BOUND'\ a\ 0) = CLAMP\ (MASTER\text{-}BOUND''\ a)$
$CLAMP\ (MASTER\text{-}BOUND'\ a\ 1) = CLAMP\ (\lambda x.\ CLAMP\ (MASTER\text{-}BOUND''\ a)\ x * CLAMP\ (\lambda x.\ ln\ (real\ x))\ x)$
$CLAMP\ (MASTER\text{-}BOUND'\ a\ (numeral\ n)) =$
$CLAMP\ (\lambda x.\ CLAMP\ (MASTER\text{-}BOUND''\ a)\ x * CLAMP\ (\lambda x.\ ln\ (real\ x)\ \hat{}\ numeral\ n)\ x)$
$CLAMP\ (MASTER\text{-}BOUND'\ a\ (-1)) =$
$CLAMP\ (\lambda x.\ CLAMP\ (MASTER\text{-}BOUND''\ a)\ x\ /\ CLAMP\ (\lambda x.\ ln\ (real\ x))\ x)$
$CLAMP\ (MASTER\text{-}BOUND'\ a\ (-numeral\ n)) =$

CLAMP (λ*x*. *CLAMP* (*MASTER-BOUND''* *a*) *x* / *CLAMP* (λ*x*. *ln* (*real* *x*) ^ *numeral* *n*) *x*)
CLAMP (*MASTER-BOUND'* *a* *b*) =
CLAMP (λ*x*. *CLAMP* (*MASTER-BOUND''* *a*) *x* ∗ *CLAMP* (λ*x*. *ln* (*real* *x*) *powr* *b*) *x*)

CLAMP (*MASTER-BOUND''* *0*) = *CLAMP* (λ*x*. *1*)
CLAMP (*MASTER-BOUND''* *1*) = *CLAMP* (λ*x*. (*real* *x*))
CLAMP (*MASTER-BOUND''* (*numeral* *n*)) = *CLAMP* (λ*x*. (*real* *x*) ^ *numeral* *n*)
CLAMP (*MASTER-BOUND''* (−*1*)) = *CLAMP* (λ*x*. *1* / (*real* *x*))
CLAMP (*MASTER-BOUND''* (−*numeral* *n*)) = *CLAMP* (λ*x*. *1* / (*real* *x*) ^ *numeral* *n*)
CLAMP (*MASTER-BOUND''* *a*) = *CLAMP* (λ*x*. (*real* *x*) *powr* *a*)

and *MASTER-BOUND-UNCLAMP*:
CLAMP (λ*x*. *CLAMP* *f* *x* ∗ *CLAMP* *g* *x*) = *CLAMP* (λ*x*. *f* *x* ∗ *g* *x*)
CLAMP (λ*x*. *CLAMP* *f* *x* / *CLAMP* *g* *x*) = *CLAMP* (λ*x*. *f* *x* / *g* *x*)
CLAMP (*CLAMP* *f*) = *CLAMP* *f*
⟨*proof*⟩

context
begin

private lemma *CLAMP-*:
landau-symbol *L* *L'* *Lr* ⟹ *L* *at-top* (*f*::*nat* ⇒ *real*) ≡ *L* *at-top* (λ*x*. *CLAMP* *f* *x*)
⟨*proof*⟩ **lemma** *UNCLAMP'-*:
landau-symbol *L* *L'* *Lr* ⟹ *L* *at-top* (*CLAMP'* (*MASTER-BOUND* *a* *b* *c*)) ≡ *L* *at-top* (*MASTER-BOUND* *a* *b* *c*)
⟨*proof*⟩ **lemma** *UNCLAMP-*:
landau-symbol *L* *L'* *Lr* ⟹ *L* *at-top* (*CLAMP* *f*) ≡ *L* *at-top* (*f*)
⟨*proof*⟩

lemmas *CLAMP* = *landau-symbols*[*THEN* *CLAMP-*]
lemmas *UNCLAMP'* = *landau-symbols*[*THEN* *UNCLAMP'-*]
lemmas *UNCLAMP* = *landau-symbols*[*THEN* *UNCLAMP-*]
end

lemma *propagate-CLAMP*:
CLAMP (λ*x*. *f* *x* ∗ *g* *x*) = *CLAMP'* (λ*x*. *CLAMP* *f* *x* ∗ *CLAMP* *g* *x*)
CLAMP (λ*x*. *f* *x* / *g* *x*) = *CLAMP'* (λ*x*. *CLAMP* *f* *x* / *CLAMP* *g* *x*)
CLAMP (λ*x*. *inverse* (*f* *x*)) = *CLAMP'* (λ*x*. *inverse* (*CLAMP* *f* *x*))
CLAMP (λ*x*. *real* *x*) = *CLAMP'* (*MASTER-BOUND* *1* *0* *0*)
CLAMP (λ*x*. *real* *x* *powr* *a*) = *CLAMP'* (*MASTER-BOUND* *a* *0* *0*)
CLAMP (λ*x*. *real* *x* ^ *a'*) = *CLAMP'* (*MASTER-BOUND* (*real* *a'*) *0* *0*)
CLAMP (λ*x*. *ln* (*real* *x*)) = *CLAMP'* (*MASTER-BOUND* *0* *1* *0*)
CLAMP (λ*x*. *ln* (*real* *x*) *powr* *b*) = *CLAMP'* (*MASTER-BOUND* *0* *b* *0*)

$CLAMP\ (\lambda x.\ ln\ (real\ x)\ \widehat{}\ b') = CLAMP'\ (MASTER\text{-}BOUND\ 0\ (real\ b')\ 0)$
$CLAMP\ (\lambda x.\ ln\ (ln\ (real\ x))) = CLAMP'\ (MASTER\text{-}BOUND\ 0\ 0\ 1)$
$CLAMP\ (\lambda x.\ ln\ (ln\ (real\ x))\ powr\ c) = CLAMP'\ (MASTER\text{-}BOUND\ 0\ 0\ c)$
$CLAMP\ (\lambda x.\ ln\ (ln\ (real\ x))\ \widehat{}\ c') = CLAMP'\ (MASTER\text{-}BOUND\ 0\ 0\ (real\ c'))$
$CLAMP'\ (CLAMP\ f) = CLAMP'\ f$
$CLAMP'\ (\lambda x.\ CLAMP'\ (MASTER\text{-}BOUND\ a1\ b1\ c1)\ x * CLAMP'\ (MASTER\text{-}BOUND\ a2\ b2\ c2)\ x) =$
$CLAMP'\ (MASTER\text{-}BOUND\ (a1+a2)\ (b1+b2)\ (c1+c2))$
$CLAMP'\ (\lambda x.\ CLAMP'\ (MASTER\text{-}BOUND\ a1\ b1\ c1)\ x\ /\ CLAMP'\ (MASTER\text{-}BOUND\ a2\ b2\ c2)\ x) =$
$CLAMP'\ (MASTER\text{-}BOUND\ (a1-a2)\ (b1-b2)\ (c1-c2))$
$CLAMP'\ (\lambda x.\ inverse\ (MASTER\text{-}BOUND\ a1\ b1\ c1\ x)) = CLAMP'\ (MASTER\text{-}BOUND\ (-a1)\ (-b1)\ (-c1))$
⟨*proof*⟩

lemma *numeral-assoc-simps*:
$((a::real) + numeral\ b) + numeral\ c = a + numeral\ (b + c)$
$(a + numeral\ b) - numeral\ c = a + neg\text{-}numeral\text{-}class.sub\ b\ c$
$(a - numeral\ b) + numeral\ c = a + neg\text{-}numeral\text{-}class.sub\ c\ b$
$(a - numeral\ b) - numeral\ c = a - numeral\ (b + c)$ ⟨*proof*⟩

lemmas *CLAMP-aux* =
arith-simps numeral-assoc-simps of-nat-power of-nat-mult of-nat-numeral one-add-one numeral-One [*symmetric*]

lemmas *CLAMP-postproc* = *numeral-One*

context *master-theorem-function*
begin

lemma *master1-bigo-automation*:
assumes $g \in O(\lambda x.\ real\ x\ powr\ p')$ $1 < (\sum i{<}k.\ as\ !\ i * bs\ !\ i\ powr\ p')$
shows $f \in O(MASTER\text{-}BOUND\ p\ 0\ 0)$
⟨*proof*⟩

lemma *master1-automation*:
assumes $g \in O(MASTER\text{-}BOUND''\ p')$ $1 < (\sum i{<}k.\ as\ !\ i * bs\ !\ i\ powr\ p')$
$eventually\ (\lambda x.\ f\ x > 0)\ at\text{-}top$
shows $f \in \Theta(MASTER\text{-}BOUND\ p\ 0\ 0)$
⟨*proof*⟩

lemma *master2-1-automation*:
assumes $g \in \Theta(MASTER\text{-}BOUND'\ p\ p')$ $p' < -1$
shows $f \in \Theta(MASTER\text{-}BOUND\ p\ 0\ 0)$
⟨*proof*⟩

lemma *master2-2-automation*:
assumes $g \in \Theta(MASTER\text{-}BOUND'\ p\ (-1))$
shows $f \in \Theta(MASTER\text{-}BOUND\ p\ 0\ 1)$

⟨*proof*⟩

lemma *master2-3-automation*:
  assumes $g \in \Theta(MASTER\text{-}BOUND'\ p\ (p' - 1))\ p' > 0$
  shows $f \in \Theta(MASTER\text{-}BOUND\ p\ p'\ 0)$
⟨*proof*⟩

lemma *master3-automation*:
  assumes $g \in \Theta(MASTER\text{-}BOUND''\ p')\ 1 > (\sum i{<}k.\ as\ !\ i * bs\ !\ i\ powr\ p')$
  shows $f \in \Theta(MASTER\text{-}BOUND\ p'\ 0\ 0)$
⟨*proof*⟩

lemmas *master-automation* =
  master1-automation master2-1-automation master2-2-automation
  master2-2-automation master3-automation

⟨*ML*⟩

end

definition *arith-consts* $(x :: real)\ (y :: nat) =$
  $(if\ \neg\ (-x) + 3\ /\ x * 5 - 1 \le x \wedge True \vee True \longrightarrow True\ then$
  $x < inverse\ 3\ powr\ 21\ else\ x = real\ (Suc\ 0\ \hat{}\ 2\ +$
  $(if\ 42 - x \le 1 \wedge 1\ div\ y = y\ mod\ 2 \vee y < Numeral1\ then\ 0\ else\ 0)) + Numeral1)$

⟨*ML*⟩

hide-const *arith-consts*

⟨*ML*⟩

hide-const *CLAMP CLAMP′ MASTER-BOUND MASTER-BOUND′ MASTER-BOUND′′*

end
theory *Akra-Bazzi-Approximation*
imports
  Complex-Main
  Akra-Bazzi-Method
  HOL−Decision-Procs.Approximation
begin

context *akra-bazzi-params-nonzero*
begin

lemma *sum-alt*: $(\sum i{<}k.\ as!i * bs!i\ powr\ p') = (\sum i{<}k.\ as!i * exp\ (p' * ln\ (bs!i)))$

⟨*proof*⟩

lemma *akra-bazzi-p-rel-intros-aux*:
$1 < (\sum i{<}k.\ as!i * exp\ (p' * ln\ (bs!i))) \Longrightarrow p' < p$
$1 > (\sum i{<}k.\ as!i * exp\ (p' * ln\ (bs!i))) \Longrightarrow p' > p$
$1 \leq (\sum i{<}k.\ as!i * exp\ (p' * ln\ (bs!i))) \Longrightarrow p' \leq p$
$1 \geq (\sum i{<}k.\ as!i * exp\ (p' * ln\ (bs!i))) \Longrightarrow p' \geq p$
$(\sum i{<}k.\ as!i * exp\ (x * ln\ (bs!i))) \leq 1 \wedge (\sum i{<}k.\ as!i * exp\ (y * ln\ (bs!i))) \geq 1 \Longrightarrow p \in \{y..x\}$
$(\sum i{<}k.\ as!i * exp\ (x * ln\ (bs!i))) < 1 \wedge (\sum i{<}k.\ as!i * exp\ (y * ln\ (bs!i))) > 1 \Longrightarrow p \in \{y{<}..{<}x\}$
⟨*proof*⟩
end

lemmas *akra-bazzi-p-rel-intros-exp* =
akra-bazzi-params-nonzero.akra-bazzi-p-rel-intros-aux[*rotated*, *OF* - *akra-bazzi-params-nonzeroI*]

lemma *eval-akra-bazzi-sum*:
$(\sum i{<}0.\ as!i * exp\ (x * ln\ (bs!i))) = 0$
$(\sum i{<}Suc\ 0.\ (a\#as)!i * exp\ (x * ln\ ((b\#bs)!i))) = a * exp\ (x * ln\ b)$
$(\sum i{<}Suc\ k.\ (a\#as)!i * exp\ (x * ln\ ((b\#bs)!i))) = a * exp\ (x * ln\ b) + (\sum i{<}k.\ as!i * exp\ (x * ln\ (bs!i)))$
⟨*proof*⟩

⟨*ML*⟩

end

8 Examples

theory *Master-Theorem-Examples*
imports
Complex-Main
Akra-Bazzi-Method
Akra-Bazzi-Approximation
begin

8.1 Merge sort

function *merge-sort-cost* :: (*nat* ⇒ *real*) ⇒ *nat* ⇒ *real* **where**
$merge\text{-}sort\text{-}cost\ t\ 0 = 0$
$|\ merge\text{-}sort\text{-}cost\ t\ 1 = 1$
$|\ n \geq 2 \Longrightarrow merge\text{-}sort\text{-}cost\ t\ n = merge\text{-}sort\text{-}cost\ t\ (nat\ \lfloor real\ n\ /\ 2\rfloor) + merge\text{-}sort\text{-}cost\ t\ (nat\ \lceil real\ n\ /\ 2\rceil) + t\ n$
⟨*proof*⟩

termination ⟨*proof*⟩

lemma *merge-sort-nonneg*[*simp*]: $(\bigwedge n.\ t\ n \geq 0) \Longrightarrow$ *merge-sort-cost t x* ≥ 0
⟨*proof*⟩

lemma $t \in \Theta(\lambda n.\ real\ n) \Longrightarrow (\bigwedge n.\ t\ n \geq 0) \Longrightarrow$ *merge-sort-cost t* $\in \Theta(\lambda n.\ real\ n * ln\ (real\ n))$
⟨*proof*⟩

8.2 Karatsuba multiplication

function *karatsuba-cost* :: *nat* ⇒ *real* **where**
karatsuba-cost 0 = *0*
| *karatsuba-cost 1* = *1*
| $n \geq 2 \Longrightarrow$ *karatsuba-cost n* =
$3 *$ *karatsuba-cost* (*nat* ⌈*real n* / *2*⌉) + *real n*
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *karatsuba-cost-nonneg*[*simp*]: *karatsuba-cost n* ≥ 0
⟨*proof*⟩

lemma *karatsuba-cost* $\in O(\lambda n.\ real\ n\ powr\ log\ 2\ 3)$
⟨*proof*⟩

lemma *karatsuba-cost-pos*: $n \geq 1 \Longrightarrow$ *karatsuba-cost n* > 0
⟨*proof*⟩

lemma *karatsuba-cost* $\in \Theta(\lambda n.\ real\ n\ powr\ log\ 2\ 3)$
⟨*proof*⟩

8.3 Strassen matrix multiplication

function *strassen-cost* :: *nat* ⇒ *real* **where**
strassen-cost 0 = *0*
| *strassen-cost 1* = *1*
| $n \geq 2 \Longrightarrow$ *strassen-cost n* = $7 *$ *strassen-cost* (*nat* ⌈*real n* / *2*⌉) + *real* (*n*^*2*)
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *strassen-cost-nonneg*[*simp*]: *strassen-cost n* ≥ 0
⟨*proof*⟩

lemma *strassen-cost* $\in O(\lambda n.\ real\ n\ powr\ log\ 2\ 7)$
⟨*proof*⟩

lemma *strassen-cost-pos*: $n \geq 1 \Longrightarrow$ *strassen-cost n* > 0
⟨*proof*⟩

lemma *strassen-cost* $\in \Theta(\lambda n.\ real\ n\ powr\ log\ 2\ 7)$

⟨*proof*⟩

8.4 Deterministic select

function *select-cost* :: *nat* ⇒ *real* **where**
n ≤ *20* ⟹ *select-cost n* = *0*
| *n* > *20* ⟹ *select-cost n* =
select-cost (*nat* ⌊*real n* / *5*⌋) + *select-cost* (*nat* ⌊*7* ∗ *real n* / *10*⌋ + *6*) + *12* ∗ *real n* / *5*
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *select-cost* ∈ Θ(λ*n*. *real n*)
⟨*proof*⟩

8.5 Decreasing function

function *dec-cost* :: *nat* ⇒ *real* **where**
n ≤ *2* ⟹ *dec-cost n* = *1*
| *n* > *2* ⟹ *dec-cost n* = *0.5*∗*dec-cost* (*nat* ⌊*real n* / *2*⌋) + *1* / *real n*
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *dec-cost* ∈ Θ(λ*x*::*nat*. *ln x* / *x*)
⟨*proof*⟩

8.6 Example taken from Drmota and Szpakowski

function *drmota1* :: *nat* ⇒ *real* **where**
n < *20* ⟹ *drmota1 n* = *1*
| *n* ≥ *20* ⟹ *drmota1 n* = *2* ∗ *drmota1* (*nat* ⌊*real n*/*2*⌋) + *8*/*9* ∗ *drmota1* (*nat* ⌊*3*∗*real n*/*4*⌋) + *real n*ˆ*2* / *ln* (*real n*)
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *drmota1* ∈ Θ(λ*n*::*real*. *n*ˆ*2* ∗ *ln* (*ln n*))
⟨*proof*⟩

function *drmota2* :: *nat* ⇒ *real* **where**
n < *20* ⟹ *drmota2 n* = *1*
| *n* ≥ *20* ⟹ *drmota2 n* = *1*/*3* ∗ *drmota2* (*nat* ⌊*real n*/*3* + *1*/*2*⌋) + *2*/*3* ∗ *drmota2* (*nat* ⌊*2*∗*real n*/*3* − *1*/*2*⌋) + *1*
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *drmota2* ∈ Θ(λ*x*. *ln* (*real x*))
⟨*proof*⟩

lemma *boncelet-phrase-length*:
fixes $p\ \delta$ :: *real* **assumes** *p*: $p > 0$ $p < 1$ **and** δ: $\delta > 0$ $\delta < 1$ $2{*}p + \delta < 2$
fixes *d* :: *nat* ⇒ *real*
defines $q \equiv 1 - p$
assumes *d-nonneg*: $\bigwedge n.\ d\ n \geq 0$
assumes *d-rec*: $\bigwedge n.\ n \geq 2 \Longrightarrow d\ n = 1 + p * d\ (nat\ \lfloor p * real\ n + \delta \rfloor) + q * d\ (nat\ \lfloor q * real\ n - \delta \rfloor)$
shows $d \in \Theta(\lambda x.\ ln\ x)$
⟨*proof*⟩

8.7 Transcendental exponents

function *foo-cost* :: *nat* ⇒ *real* **where**
$n < 200 \Longrightarrow$ *foo-cost* $n = 0$
| $n \geq 200 \Longrightarrow$ *foo-cost* $n =$
foo-cost $(nat\ \lfloor real\ n\ /\ 3 \rfloor)$ + *foo-cost* $(nat\ \lfloor 3 * real\ n\ /\ 4 \rfloor + 42) + real\ n$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *foo-cost-nonneg* [*simp*]: *foo-cost* $n \geq 0$
⟨*proof*⟩

lemma *foo-cost* $\in \Theta(\lambda n.\ real\ n\ powr$ *akra-bazzi-exponent* $[1,1]\ [1/3,3/4])$
⟨*proof*⟩

lemma *akra-bazzi-exponent* $[1,1]\ [1/3,3/4] \in \{1.1519623..1.1519624\}$
⟨*proof*⟩

8.8 Functions in locale contexts

locale *det-select* =
fixes *b* :: *real*
assumes *b*: $b > 0$ $b < 7/10$
begin

function *select-cost'* :: *nat* ⇒ *real* **where**
$n \leq 20 \Longrightarrow$ *select-cost'* $n = 0$
| $n > 20 \Longrightarrow$ *select-cost'* $n =$
select-cost' $(nat\ \lfloor real\ n\ /\ 5 \rfloor)$ + *select-cost'* $(nat\ \lfloor b * real\ n \rfloor + 6) + 6 * real\ n + 5$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma $a \geq 0 \Longrightarrow$ *select-cost'* $\in \Theta(\lambda n.\ real\ n)$
⟨*proof*⟩

end

8.9 Non-curried functions

function *baz-cost* :: *nat* × *nat* ⇒ *real* **where**
$n \leq 2 \Longrightarrow$ *baz-cost* $(a, n) = 0$
| $n > 2 \Longrightarrow$ *baz-cost* $(a, n) = 3 * $ *baz-cost* $(a, \mathit{nat}\ \lfloor \mathit{real}\ n\ /\ 2 \rfloor) + \mathit{real}\ a$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *baz-cost-nonneg* [*simp*]: $a \geq 0 \Longrightarrow$ *baz-cost* $(a, n) \geq 0$
⟨*proof*⟩

lemma
assumes $a > 0$
shows $(\lambda x.\ \mathit{baz\text{-}cost}\ (a, x)) \in \Theta(\lambda x.\ x\ \mathit{powr}\ \mathit{log}\ 2\ 3)$
⟨*proof*⟩

function *bar-cost* :: *nat* × *nat* ⇒ *real* **where**
$n \leq 2 \Longrightarrow$ *bar-cost* $(a, n) = 0$
| $n > 2 \Longrightarrow$ *bar-cost* $(a, n) = 3 * $ *bar-cost* $(2 * a, \mathit{nat}\ \lfloor \mathit{real}\ n\ /\ 2 \rfloor) + \mathit{real}\ a$
⟨*proof*⟩
termination ⟨*proof*⟩

8.10 Ham-sandwich trees

function *ham-sandwich-cost* :: *nat* ⇒ *real* **where**
$n < 4 \Longrightarrow$ *ham-sandwich-cost* $n = 1$
| $n \geq 4 \Longrightarrow$ *ham-sandwich-cost* $n =$
ham-sandwich-cost $(\mathit{nat}\ \lfloor n/4 \rfloor)$ + *ham-sandwich-cost* $(\mathit{nat}\ \lfloor n/2 \rfloor) + 1$
⟨*proof*⟩
termination ⟨*proof*⟩

lemma *ham-sandwich-cost-pos* [*simp*]: *ham-sandwich-cost* $n > 0$
⟨*proof*⟩

The golden ratio

definition $\varphi = ((1 + \mathit{sqrt}\ 5)\ /\ 2 :: \mathit{real})$

lemma *φ-pos* [*simp*]: $\varphi > 0$ **and** *φ-nonneg* [*simp*]: $\varphi \geq 0$ **and** *φ-nonzero* [*simp*]:
$\varphi \neq 0$
⟨*proof*⟩

lemma *ham-sandwich-cost* $\in \Theta(\lambda n.\ n\ \mathit{powr}\ (\mathit{log}\ 2\ \varphi))$
⟨*proof*⟩

end

References

[1] M. Akra and L. Bazzi. On the solution of linear recurrence equations. *Computational Optimization and Applications*, 10(2):195–210, 1998.

[2] T. Leighton. Notes on better Master theorems for divide-and-conquer recurrences. 1996.