Bottom-Up Computation Using Trees of Sublists: Proofs

Shin-Cheng Mu

Institute of Information Science, Academia Sinica

Abstract

Proofs accompanying the paper Bottom-up computation using trees of sublists, *Journal of Functional Programming Special Issue on Program Calculation*, 2025.

1 DEFINITIONS

Non-dependently typed version of the binomial tree:

$$\mathbf{data}\ \mathsf{B}\ a = \mathsf{T}\ a \mid \mathsf{N}\ (\mathsf{B}\ a)\ (\mathsf{B}\ a) \quad ,$$

equipped with its *map* and *zip*:

$$\begin{aligned}
&B &&:: (a \to b) \to \mathsf{B}\ a \to \mathsf{B}\ b \quad ,\\
&zipBW &&:: (a \to b \to c) \to \mathsf{B}\ a \to \mathsf{B}\ b \to \mathsf{B}\ c \quad .
\end{aligned}$$

And we have $unT\ (\mathsf{T}\ x) = x$.

In fact, we do not generate all trees. The shapes of trees we generate correspond to the function ch, a tree version of the function that chooses a given number of elements from a list: The shape

$$\begin{aligned}
&ch :: \mathbb{N} \to \mathsf{L}\ a \to \mathsf{B}\ (\mathsf{L}\ a)\\
&ch\ 0 \quad\ \ _ \qquad\ \ = \mathsf{T}\ [\,]\\
&ch\ k \quad\ \ xs \qquad \mid k == length\ xs = \mathsf{T}\ xs\\
&ch\ (1+k)\ (x:xs) = \mathsf{N}\ (B\ (x:)\ (ch\ k\ xs))\ (ch\ (1+k)\ xs) \quad .
\end{aligned}$$

We may also constraint the shape of the trees by dependent type:

$$\begin{aligned}
&\mathbf{data}\ \mathsf{B}\ (a : \mathsf{Set}) : \mathbb{N} \to \mathbb{N} \to \mathsf{Set}\ \mathbf{where}\\
&\quad \mathsf{T}_0 : a \to \mathsf{B}\ a\ 0\ n\\
&\quad \mathsf{T}_n : a \to \mathsf{B}\ a\ (suc\ n)\ (suc\ n)\\
&\quad \mathsf{N}\ \ : \mathsf{B}\ a\ k\ n \to \mathsf{B}\ a\ (suc\ k)\ n \to \mathsf{B}\ a\ (suc\ k)\ (suc\ n) \quad .
\end{aligned}$$

For this note we will stay in the non-dependently typed world.

2 UPGRADE

The paper derived the following function up:

$$\begin{aligned}
&up :: \mathsf{B}\ a \to \mathsf{B}\ (\mathsf{L}\ a)\\
&up\ (\mathsf{N}\ (\mathsf{T}\ p)\ (\mathsf{T}\ q)) = \mathsf{T}\ [p,q]\\
&up\ (\mathsf{N}\ t \qquad (\mathsf{T}\ q)) = \mathsf{T}\ (unT\ (up\ t) +\!\!+ [q])\\
&up\ (\mathsf{N}\ (\mathsf{T}\ p)\ u \qquad) = \mathsf{N}\ (B\ (\lambda q \to [p,q])\ u)\ (up\ u)\\
&up\ (\mathsf{N}\ t \qquad u \qquad) = \mathsf{N}\ (zipBW\ snoc\ (up\ t)\ u)\ (up\ u) \quad .
\end{aligned}$$

With dependent type, up could be typed

$$up : \forall \{a\ k\ n\} \to (0 < k) \to (k < n) \to \mathsf{B}\ a\ k\ n \to \mathsf{B}\ (\mathsf{Vec}\ a\ (1+k))\ (1+k)\ n \quad ,$$

but again, we stay in the non-dependent realm in this note.

The derivation of up was driven by trying to prove the following theorem:

Theorem 1.

$$\begin{aligned} &(\forall xs, k : 2 \leqslant 1+k \leqslant length\ xs\ : \\ &\quad up\ (ch\ k\ xs) = B\ subs\ (ch\ (1+k)\ xs))\quad . \end{aligned} \tag{1}$$

And here is the constructed proof.

Proof. The proof is is an induction on xs. The case analysis follows the shape of $ch\ (1+k)\ xs$ (on the RHS of (1)). Therefore, there is a base case, a case when xs is non-empty and $1+k = length\ xs$, and a case when $1+k < length\ xs$. However, since the constraints demand that xs has at least two elements, the base case will be lists of length 2, and in the inductive cases the length of the list will be at least 3.

CASE 1. $xs := [y, z]$.
The constraints force k to be 1. We reason:

$$\begin{aligned} & B\ subs\ (ch\ 2\ [y,z]) \\ = &\quad \{ \text{def. of } ch \} \\ & B\ subs\ (\mathsf{T}\ [y,z]) \\ = &\quad \{ \text{def. of } B \text{ and } subs \} \\ & \mathsf{T}\ [[y],[z]] \\ = &\quad \{ \text{definition of } up \} \\ & up\ (\mathsf{N}\ (\mathsf{T}\ [y])\ (\mathsf{T}\ [z])) \\ = &\quad \{ \text{def. of } ch \} \\ & up\ (ch\ 1\ [y,z])\quad . \end{aligned}$$

case 2. $xs := x : xs$, $k := 1+k$, where $length\ xs \geqslant 2$, and $1 + (1+k) = length\ (x : xs)$.

$$\begin{aligned} & up\ (ch\ (1+k)\ (x : xs)) \\ = &\quad \{ \text{def. of } ch\text{, since } 1+k < length\ (x:xs) \} \\ & up\ (\mathsf{N}\ (B\ (x{:})\ (ch\ k\ xs))\ (ch\ (1+k)\ xs)) \\ = &\quad \{ \text{def. of } ch\text{, since } 1+k = length\ xs \} \\ & up\ (\mathsf{N}\ (B\ (x{:})\ (ch\ k\ xs))\ (\mathsf{T}\ xs)) \\ = &\quad \{ \text{def. of } up \} \\ & \mathsf{T}\ (unT\ (up\ (B\ (x{:})\ (ch\ k\ xs))) \mathbin{+\!\!+} [xs]) \\ = &\quad \{ up \text{ natural} \} \\ & \mathsf{T}\ (unT\ (B\ (L\ (x{:}))\ (up\ (ch\ k\ xs))) \mathbin{+\!\!+} [xs]) \\ = &\quad \{ \text{induction} \} \\ & \mathsf{T}\ (unT\ (B\ (L\ (x{:}))\ (B\ subs\ (ch\ (1+k)\ xs))) \mathbin{+\!\!+} [xs]) \\ = &\quad \{ \text{def. of } ch\text{, since } 1+k = length\ xs \} \\ & \mathsf{T}\ (unT\ (B\ (L\ (x{:}))\ (B\ subs\ (\mathsf{T}\ xs))) \mathbin{+\!\!+} [xs]) \\ = &\quad \{ \text{def. of } B \text{ and } L \} \\ & \mathsf{T}\ (L\ (x{:})\ (subs\ xs) \mathbin{+\!\!+} [xs]) \\ = &\quad \{ \text{def. of } subs \} \\ & \mathsf{T}\ (subs\ (x : xs)) \\ = &\quad \{ \text{def. of } B \} \\ & B\ subs\ (\mathsf{T}\ (x : xs)) \\ = &\quad \{ \text{def. of } ch\text{, since } 2+k = length\ (x:xs) \} \\ & B\ subs\ (ch\ (2+k)\ (x : xs))\quad . \end{aligned}$$

case 3. $xs := x : xs$, $k := 1 + k$, where $length\ xs \geqslant 2$, and $1 + (1 + k) < length\ (x : xs)$. The constraints become $2 \leqslant 2 + k < length\ (x : xs)$. The property to prove is:

$$up\ (ch\ (1+k)\ (x : xs)) = B\ subs\ (ch\ (2+k)\ (x : xs))\ \ .$$

We split this into two sub-cases:

case 3.1 $k := 0$.

$$\begin{array}{ll}
 & up\ (ch\ 1\ (x : xs)) \\
= & \quad \{\ \text{def. of } ch\text{, since } 1 < length\ (x : xs)\ \} \\
 & up\ (\mathsf{N}\ (B\ (x{:})\ (ch\ 0\ xs))\ (ch\ 1\ xs)) \\
= & \quad \{\ \text{def. of } ch\ \} \\
 & up\ (\mathsf{N}\ (\mathsf{T}\ [x])\ (ch\ 1\ xs)) \\
= & \quad \{\ \text{def. of } up\ \} \\
 & \mathsf{N}\ (B\ (\lambda q \rightarrow [[x], q])\ (ch\ 1\ xs))\ (up\ (ch\ 1\ xs)) \\
= & \quad \{\ (*)\text{ see below }\} \\
 & \mathsf{N}\ (B\ (subs \circ (x{:}))\ (ch\ 1\ xs))\ (up\ (ch\ 1\ xs)) \\
= & \quad \{\ \text{induction}\ \} \\
 & \mathsf{N}\ (B\ (subs \circ (x{:}))\ (ch\ 1\ xs))\ (B\ subs\ (ch\ 2\ xs)) \\
= & \quad \{\ \text{def. of } B\ \} \\
 & B\ subs\ (\mathsf{N}\ (B\ (x{:})\ (ch\ 1\ xs))\ (ch\ 2\ xs)) \\
= & \quad \{\ \text{def. of } ch\text{, since } 2 < length\ (x : xs)\ \} \\
 & B\ subs\ (ch\ 2\ (x : xs))\ \ .
\end{array}$$

The step (*) holds because every tip in $ch\ 1\ xs$ is a singleton list, and for a singleton list $[z]$, we have $subs\ (x : [z]) = [[x], [z]]$.

case 3.2 $0 < k$ (and $k < length\ xs - 1$). For this case we need the following auxiliary properties. Recall that

- by definition, $sub\ (x : xs) = L\ (x{:})\ (sub\ xs) +\!\!+ [xs]$.
- Given a tree u and functions f, g, and h, by naturality of $zipBW$ we have:

$$B\ (\lambda z \rightarrow f\ (g\ z)\ (h\ z))\ u = zipBW\ f\ (B\ g\ u)\ (B\ h\ u)\ \ . \tag{2}$$

- Therefore, letting $g = L\ (x{:}) \circ subs$, $h = id$, and $f = snoc$ in (2), where $snoc\ ys\ z = ys +\!\!+ [z]$, we have:

$$B\ (subs \circ (x{:}))\ u = zipBW\ snoc\ (B\ (L\ (x{:}) \circ subs)\ u)\ u\ \ . \tag{3}$$

We reason:

$$\begin{array}{ll}
 & up\ (ch\ (1+k)\ (x : xs)) \\
= & \quad \{\ \text{def. of } ch\text{, since } 1 + k < length\ (x : xs)\ \} \\
 & up\ (\mathsf{N}\ (B\ (x{:})\ (ch\ k\ xs))\ (ch\ (1+k)\ xs))\ \ . \\
= & \quad \{\ \text{def. of } up\text{, since } k \neq 0 \text{ and } 1 + k < length\ xs\ \} \\
 & \mathsf{N}\ (zipBW\ snoc\ (up \circ B\ (x{:}) \circ ch\ k\ \$\ xs)\ (ch\ (1+k)\ xs)) \\
 & \quad (up\ (ch\ (1+k)\ xs))
\end{array}$$

Let us focus on the first argument to N:

$$\begin{array}{ll}
 & zipBW\ snoc\ (up \circ B\ (x{:}) \circ ch\ k\ \$\ xs)\ (ch\ (1+k)\ xs) \\
= & \quad \{\ up\ \text{natural}\ \}
\end{array}$$

$$
\begin{aligned}
& \mathit{zipBW}\ \mathit{snoc}\ (B\ (L\ (x{:}))\circ \mathit{up}\circ \mathit{ch}\ k\ \$\ \mathit{xs})\ (\mathit{ch}\ (1+k)\ \mathit{xs})\\
=\ & \quad \{\ \text{induction}\ \}\\
& \mathit{zipBW}\ \mathit{snoc}\ (B\ (L\ (x{:})\circ \mathit{subs})\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\ (\mathit{ch}\ (1+k)\ \mathit{xs})\\
=\ & \quad \{\ \text{by (3)}\ \}\\
& B\ (\mathit{subs}\circ (x{:}))\ (\mathit{ch}\ (1+k)\ \mathit{xs})\quad .
\end{aligned}
$$

We continue:

$$
\begin{aligned}
& \mathsf{N}\ (\mathit{zipBW}\ \mathit{snoc}\ (\mathit{up}\circ B\ (x{:})\circ \mathit{ch}\ k\ \$\ \mathit{xs})\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\\
& \quad (\mathit{up}\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\\
=\ & \quad \{\ \text{calculation above}\ \}\\
& \mathsf{N}\ (B\ (\mathit{subs}\circ (x{:}))\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\ (\mathit{up}\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\\
=\ & \quad \{\ \text{induction}\ \}\\
& \mathsf{N}\ (B\ (\mathit{subs}\circ (x{:}))\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\ (B\ \mathit{subs}\ (\mathit{ch}\ (2+k)\ \mathit{xs}))\\
=\ & \quad \{\ \text{def. of } B\ \}\\
& B\ \mathit{subs}\ (\mathsf{N}\ (B\ (x{:})\ (\mathit{ch}\ (1+k)\ \mathit{xs}))\ (\mathit{ch}\ (2+k)\ \mathit{xs}))\\
=\ & \quad \{\ \text{def. of } \mathit{ch}\ \}\\
& B\ \mathit{subs}\ (\mathit{ch}\ (2+k)\ (x:\mathit{xs}))\quad .
\end{aligned}
$$

□

3 TOP-DOWN AND BOTTOM-UP ALGORITHMS

The generic top-down algorithm is defined by:

$$
\begin{aligned}
& \mathit{td} :: \mathbb{N} \to \mathsf{L}\ \mathsf{X} \to \mathsf{Y}\\
& \mathit{td}\ 0 \qquad\ = f \circ \mathit{ex}\\
& \mathit{td}\ (1+n) = g \circ L\ (\mathit{td}\ n) \circ \mathit{subs}\quad .
\end{aligned}
$$

The intention is that $\mathit{td}\ n$ is a function defined for inputs of length exactly $1+n$.

It helps to define a variation:

$$
\begin{aligned}
& \mathit{td}' :: \mathbb{N} \to \mathsf{L}\ \mathsf{Y} \to \mathsf{Y}\\
& \mathit{td}'\ 0 \qquad\ = \mathit{ex}\\
& \mathit{td}'\ (1+n) = g \circ L\ (\mathit{td}'\ n) \circ \mathit{subs}\quad .
\end{aligned}
$$

The difference is that td' calls only ex in the base case. It is a routine induction showing that

$$\mathit{td}\ n = \mathit{td}'\ n \circ L\ f\quad . \tag{4}$$

All the calls to f are thus factored to the beginning of the algorithm. We may then focus on transforming td'.

Note that for $\mathit{ch}\ n\ \mathit{xs}$ where $n = \mathit{length}\ \mathit{xs}$ always results in $\mathsf{T}\ \mathit{xs}$. That is, we have

$$\mathit{unT}\ (\mathit{ch}\ n\ \mathit{xs}) = \mathit{xs}\quad \text{, where } n = \mathit{length}\ \mathit{xs}. \tag{5}$$

Our main theorem is that

Theorem 2. *For all $n :: \mathbb{N}$ we have $\mathit{td}\ n = \mathit{bu}\ n$, where*

$$\mathit{bu}\ n = \mathit{unT} \circ (B\ g \circ \mathit{up})^n \circ B\ \mathit{ex} \circ \mathit{ch}\ 1 \circ L\ f\quad .$$

That is, the top-down algorithm $\mathit{td}\ n$ is equivalent to a bottom-up algorithm $\mathit{bu}\ n$, where the input is preprocessed by $B\ \mathit{ex} \circ \mathit{ch}\ 1 \circ L\ f$, followed by n steps of $B\ g \circ \mathit{up}$. By then we will get a singleton tree, whose content can be extracted by unT.

Proof. Let $length\ xs = 1 + n$. We reason:

$$
\begin{aligned}
& td\ n\ xs \\
= & \quad \{ \text{by (4)} \} \\
& (td'\ n \circ L\ f)\ xs \\
= & \quad \{ \text{by (5)} \} \\
& (td'\ n \circ unT \circ ch\ (1+n) \circ L\ f)\ xs \\
= & \quad \{ \text{naturality of } unT \} \\
& (unT \circ B\ (td'\ n) \circ ch\ (1+n) \circ L\ f)\ xs \\
= & \quad \{ \text{Lemma 1} \} \\
& (unT \circ (B\ g \circ up)^n \circ B\ ex \circ ch\ 1 \circ L\ f)\ xs \\
= & \quad \{ \text{definition of } bu \} \\
& bu\ n\ xs \quad .
\end{aligned}
$$

□

Lemma 1, showing that $B\ (td'\ n) \circ ch\ (1+n)$ can be performed by n steps of $B\ g \circ up$, after some preprocessing, is where the main proof is done. This is the key lemma that relates (1) to the main algorithm.

Lemma 1. $B\ (td'\ n) \circ ch\ (1+n) = (B\ g \circ up)^n \circ B\ ex \circ ch\ 1$.

Proof. For $n := 0$ both sides simplify to $B\ ex \circ ch\ 1$. For $n := 1 + n$:

$$
\begin{aligned}
& B\ (td'\ (1+n)) \circ ch\ (2+n) \\
= & \quad \{ \text{def. of } td' \} \\
& B\ (g \circ L\ (td'\ n) \circ subs) \circ ch\ (2+n) \\
= & \quad \{ \text{by (1)} \} \\
& B\ (g \circ L\ (td'\ n)) \circ up \circ ch\ (1+n) \\
= & \quad \{ up \text{ natural} \} \\
& B\ g \circ up \circ B\ (td'\ n) \circ ch\ (1+n) \\
= & \quad \{ \text{induction} \} \\
& B\ g \circ up \circ (B\ g \circ up)^n \circ B\ ex \circ ch\ 1 \\
= & \quad \{ (\circ) \text{ associative, def. of } f^n \} \\
& (B\ g \circ up)^{1+n} \circ B\ ex \circ ch\ 1 \quad .
\end{aligned}
$$

□