Parallel Expression Evaluation

Eg. Input

Possible Outputs Root value, all subvalues

Simple Alg

1) Assign a processor to each node.

2) While tree not empty do

3) If leaf "send" value to parent &

4) delete node [RAKE]

5) else if node has values then evaluate.
Bad Case for Simple Alg

Recall: Horner's Rule Polynomial eval

Input: polynomial $a_0 + a_1x + \cdots + a_nx^n$

Alg $a_0 + x(a_1 + x(\cdots + x(a_{n-1} + x_a)\cdots))$

As a tree

\[
\begin{array}{cccc}
& + & \times & \\
& & & \\
& & & \\
& a_0 & & \\
& & & \\
& & & \\
& & & \\
\end{array}
\]

Simple Alg is $O(n)$ time

$O(n^3)$ proc. time
Keeping nodes with only one value busy!

Here we view the tree edges as transformers.

Init: The edge are the identity.

\[ a \cdot y \quad \Rightarrow \quad y + b \]

\[ a \cdot y \]

\[ a \cdot (y + b) = a \cdot y + a \cdot b \]
The general case.

\[ f(y) = ay + b \quad g(y) = cy + d \]

\[ f(g(y)) = a(cy + d) + b = acy + (ad + b) \]

Note: functions \( ay + b \) are closed under compositions

We can also remove an independent set of 1-child nodes (degree 2 nodes)

Very similar to pivoting in Gaussian Elim! 
**Def** \( V_0 \ldots V_k \) is a **chain** if:

1) \( V_{i+1} \) is only child of \( V_i \), \( 0 \leq i < k \).
2) \( V_k \) has only one child & it is not a leaf.

---

**Example**

![Diagram of a chain](attachment:chain_diagram.png)

---

The **Independent Set**

1) All leaves
2) Max independent set from each maximal chain
Parallel Tree Contraction

RAKE \equiv \text{remove all leaves}

COMPRESS \equiv \text{replace each maximal chain of length } K \text{ with one of length } \frac{K}{2}.

\text{CONTRACT} = \{ \text{RAKE, COMPRESS} \}

\text{Thm} \quad \left| \text{CONTRACT}(T) \right| \leq \frac{2}{3} |T|

\text{pf} \quad \text{Def } V_0 = \text{leaves of } T

V_1 \subseteq V \text{ with 2 child}

V_2 \subseteq V \text{ with } 2 \leq \# \text{ children}

C \subseteq V_1 \text{ with child in } V_0
Claim $|V_0| > |V_a|$

Proof induct on size of $T$.

Claim: $|V_0| \geq |C|$.

Def $\quad R_a = V_0 \cup V_2 \setminus C$.

$\quad C_{om} = V_1 - R_a$.

$\quad \text{RAKE}(R_a) \subseteq V_2 \cup C \Rightarrow |\text{RAKE}(R_a)| \leq \frac{3}{2} |R_a|$.

Note $\quad C_{om} = \text{union of maximal chains}$.

$|\text{COMPRESS}(C_{om})| \leq \frac{1}{2} |C_{om}|$.

Cor After $\log_{3/2} n$ CONTRACTS two

empty
Max Value in Subtree

Input: Rooted binary tree $T$ with node weights.

Output: Max value in each subtree.

Unary fcn: $M_a(x) = \max \{ a, x \}$

Alg:

For each edge set edge $a$ transformer to $M_{-\infty}(x)$

Contraction Phase:

Base: $V(P) = M_a(V(\text{leaf}))$

$\beta \in M_a(\alpha)$

Compress

$M_b \Rightarrow M_{(a,b,V(P))}$
Expansion Phase

\[ \text{Set } \beta = M_a(\alpha) \]

---

**Prob:** Find Max of Ancestors

**Contraction**
\[ \text{Rake } = \emptyset \]
\[ \text{Compress } V(C) = \max \{ V(P), V(C) \} \]

**Expansion**
\[ \text{Rake } V(C) = \max \{ V(P), V(C) \} \]
\[ \text{Compress } V(P) = \max \{ V(GP), V(P) \} \]
Computing Low Point Numbers

Input: DFS tree of undirected graph.
Output: Low Point Numbers

Alg: T

1) Compute preorder numbers
2) Replace preorder numbers with back edge numbers
3) Compute min value in each subtree
Inorder

(backedge) 
(numbers) 

min value in subtree 
= how point.
**Lowest Common Ancestor Prob LCA**

**Input:** Rooted tree $T$ and nontree edges $e_1, \ldots, e_m$

**Output:** For each edge $e_i = (u,v)$ the LCA of $u$ & $v$.

For example:

- $LCA(V_1, V_5) = V_2$
- $LCA(V_5, V_3) = V_3$
- $LCA(V_3, V_5) = V_3$

**Note:** After an Euler Tour we get a constant work ancestor test.

$LCA(T, e_1, \ldots, e_m)$

For each round of PTC do the following:

1) *(RAKE)* Inll all leaf nodes $V$

   a) If $e$ is self loop at $V$ then $LCA(e) = V$ & delete

   b) else move edges to parent & delete leaf

2) *(Compress)* Inll for each node $V$ to be spliced out

   1) All edge $e = (v,w)$

      If $V$ ancestor of $W$ then

      $LCA(e) = V$ & delete $e$.

      else $e = (parent(V), W)$

      Splice Out $V$. 
Work and Time Efficient Tree Contraction

**Idea 1**
Do regular RAKE.
Use Random-Mate to COMPRESS chains.

**Using Chernoff Bounds**

**Thm** Randomized Tree Contraction runs in $O(\log n)$ with high prob.

Thus $W = O(n \log n)$  Time = $O(1/\log n)$.

**Idea 2**
1) Break tree into $n/\log n$ pieces each of size $\leq \log n$
2) Contract pieces to constant size.
3) Run Rand Tree Contraction on tree of size $O(n/\log n)$
A Tree into Bridges

Let $T$ be a rooted tree $T=(V,E)$.

Def: A subtree $B$ is a bridge if at most 2 attachments: a root, a leaf.

Ex: 1) Single edge
2) Induced subtree
3) [Diagram of a tree with a bridge]

Thm: For every decomposition of $T$ into $O(n/m)$ bridges of size at most $m$. 
**m-critical nodes**

\[ T = (V, E) \quad W(v) = \# \text{nodes in subtree rooted at } v. \]

**Def** \( v \) is **m-critical** if

1) \( v \) not a leaf.

2) \( \frac{rW(v)}{m} > \frac{rW(v')}{m} \quad \forall v' \in \text{children}(v). \)

**eg** 5-critical

5-bridges

Claim (see chap 3) \((m-1)\)-bridges proves thm.
Thm Tree Contraction can be done in 
$O(n)$ work/$(PT)O(\log n)$ time with high prob.

Known:Det in some bounds.

Alg 1) Compute $\log n$-critical nodes using Euler tour
2) Contract bridges
3) Contract $\frac{\log n}{\sqrt{n}}$ tree using random mate
4) Expand.