Parallel Expression Evaluation

Example

```
  +
 /\  
+  x
/\  /
+ x  
/\  /
1 4 5 3
```

Output: Value, all subvalues

Goal: Parallel Alg

Simple Alg

1) Assign a processor to each node.

While tree non empty do

2) if leaf "send" value to parent
   delete node (RAKE)

3) if node has 2 values then evaluate.
Worst Case for simple Alg

Recall: Horner's Rule

Input: polynomial \( a_0 + a_1x + a_2x^2 + \ldots + a_nx^n \)

Alg: \[ a_0 + x(a_1 + x(\ldots + x(a_{n-1} + x(a_n)) \ldots ) \]

As a tree

\[ a_0 \]
\[ + \]
\[ x \]
\[ + \]
\[ a_1 \]
\[ x \]
\[ + \]
\[ a_2 \]
\[ x \]
\[ + \]
\[ a_{n-1} \]
\[ x \]
\[ + \]
\[ a_n \]

Simple Alg

\( O(n) \) Time

\( O(n^2) \) P.T.
Keeping nodes with only one value busy!

Here we view the tree edges as transformers.

Init: The edge are the identity.

\[ f(x) + b = y + b \]

\[ a \cdot f(y) = a \cdot y \]

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

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

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

Note: fans of \( ay + b \) 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.

The Independent Set

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

RAKE = remove all leaves

COMPRESS = replace each maximal chain of length $k$ with one of length $\lfloor \frac{k}{2} \rfloor$.

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

Thm \[ |\text{CONTRACT}(T)| \leq \frac{2}{3} |T| \]

proof: Def \[ V_0 = \text{leaves of } T \]
\[ V_1 \subseteq V \text{ with 1 child} \]
\[ V_2 \subseteq V \text{ with } 2 \leq \# \text{ children} \]
\[ C \subseteq V_1 \text{ with child in } V_0 \]
\textbf{Claim:} \( |V_0| > |V_a| \)

Prove induct on size of \( T \).

\textbf{Claim:} \( |V_0| \geq |C| \)

\textbf{Def:} \( R_a = V_0 \cup V_2 \cup C \)

\( C_{om} = V_1 - R_a \)

\( \text{Rake}(R_a) \subseteq V_a \cup C \Rightarrow |\text{Rake}(R_a)| \leq \frac{2}{3} |R_a| \)

\textbf{Note:} \( C_{om} = \text{union of maximal chains} \)

\( |\text{Compress}(C_{om})| \leq \frac{1}{a} |C_{om}| \)

\textbf{Cor:} After \( \log_a n \) \( \text{CONTRACTS} \) is empty.
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 = $O(\log n)$

**Idea 2**
1) Break tree into $n/\log n$ pieces each of size $\leq \log n$
2) Contract pieces to constant size
3) Run Random 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) 

Thm. \( \forall m \exists \text{ decomp of } T \text{ into } O(n/m) \text{ bridges of size at most } m \).
\[ T = (V, E) \quad W(v) = \# \text{nodes in subtree rooted at } V_0. \]

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

1. \( V \) is not a leaf.
2. \( \frac{rw(v)}{m} > \frac{rw(v')}{m} \quad \forall v' \in \text{children}(v). \)

**Ex** \( 5 \)-critical

---

Claim (see chap 3) \((m-1)\)-bridges proves thm.
Thin Tree Contraction can be done in $O(n)$ work, $O(\log n)$ time, with high prob.
known: Det in same bounds.

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

Rooted binary tree $T$ with node weights

Goal: For each node compute max value in subtree

<table>
<thead>
<tr>
<th>Rule</th>
<th>Contract</th>
<th>Expand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compress</td>
<td>$V(P) = \max{V(P), V(N)}$</td>
<td>$V(N) = \max{V(N), V(C)}$</td>
</tr>
</tbody>
</table>
Goal: Max of ancestors

<table>
<thead>
<tr>
<th>Contract</th>
<th>Expand</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varnothing$</td>
<td>$V(L) = \max{V(L), V(P)}$</td>
</tr>
<tr>
<td>$V(C) = \max{V(C), V(N)}$</td>
<td>$V(N) = \max{V(N), V(P)}$</td>
</tr>
</tbody>
</table>

Diagram:

```
R
Comp

P ← N ← C

P ← C
```
Low Point Numbers

After computing min value of backedges per node, the problem reduces to min value of each subtree.
Lowest Common Ancestor Problem (LCA)

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

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

```
LCA(v_1, v_3) = v_2
LCA(v_3, v_5) = v_3
LCA(v_4, v_7) = v_4
```

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

Run PTC on $T$ for each round do

- for each edge $e = (e_v, e_w)$

- for each $(u, u') \in \{(v, w), (w, v)\}$

  - if $u$ ancestor $u'$ then $\text{LCA}(e) = u$

  else $e_u = \text{parent}(u)$ & $e_{u'} = \text{parent}(u')$