$\boxed{\text{Lec 5}}$ Sequences

Recall dependence graph & pebbel game

Greedy strat take at most $\frac{W}{P} + S$

Well then at each step we either :
- contribute to $\frac{W}{P}$ term
- contribute to S term
- contribute to both

So we fill $\frac{W}{P} + S$ by greedy scheduling

Work span trade off

→ Which to optimise?

$\frac{W}{P} + S$ ... usually W first. Usually give up no more than $O(\log n)$ work for better span

Array seqs

- Primitives $A[i]$, alloc, $|A|$, parFor
- Assume parFor forks as many as it wants

append A B = tab (fn i ⇒ if $i < |A|$ then $A[i]$ else $B[i - |A|]$) $(|A| + |B|)$

$W = O(|A| + |B|)$ $S = O(1)$

subseq -- tabulate and grab indicies ?

$W = \cancel{O(l)}$ $S = O(1)$

Nope spec says $O(1)$.
Because values not mutable we can reference subseq

Efficient subseq & split mid

type α seq = (α array * start * end)

→ Then operation does index manipulation without necessarily copying part of the α array.

iterate, iteratePrefixes, reduce, scan

It's just foldl

iterate : $(\beta \times \alpha \to \beta) \to \beta \to \alpha\ seq \to \beta$

$\quad\quad\quad\quad F \quad\quad\quad\quad init \quad\quad A$

$$W = O\Big(\cancel{\leqq} \sum_{i=0}^{n-1} W(f(x_i, A[i]))\Big) \qquad S = W$$

Prof's new symbol, whoops

Consider :

```
x = <init>
B = alloc |A|
for i in 0..(n-1)
    B[i] = x
    x = f(x, A[i])
ret (B, x)
```

iteratePrefixes : $(\beta \times \alpha \to \beta) \to \beta \to \alpha\ seq \to (\beta\ seq \cdot \beta)$

But if f associative and <init> is left identity of F, we can do things in parallel

$\Rightarrow$ iterate f I A $\equiv$ reduce f I A

<u>Associative funcs</u>

$+,\ *,\ \wedge,\ \ldots$

$f((l_1, r_1), (l_2, r_2)) =$ if $(r_2 > l_2)$ then $(l_1,\ r_1 - l_2 + r_2)$
else $(l_1 - r_1 + l_2,\ r_2)$

copy(x, y) = case y of NONE $\Rightarrow x$
| _ $\Rightarrow y$

<u>Examples</u>

Assuming $W_{merge} = O(n)$ $S_{merge} = O(\log n)$

iterate (merge <) ⟨⟩ ⟨⟨x⟩ : x ∈ A⟩ ← insertion sort $\quad W = O(n^2)$, $S = O(n \log n)$

reduce (merge <) ⟨⟩ ⟨⟨x⟩ : x ∈ A⟩ ← merge sort $\quad W = O(n \log n)$, $S = O(\log^2 n)$