15-210: Parallel and Sequential Data Structures and Algorithms

Costs for Sequences, Sets and Tables

1 Function Costs

ArraySequence	*Work*	*Span*
`append` (S_1, S_2)	$\|S_1\| + \|S_2\|$	1
`argmax` f S	$\|S\|$	$\log \|S\|$
`collate` f (S_1, S_2)	$\|S_1\| + \|S_2\|$	$\log(\min(\|S_1\|, \|S_2\|))$
`collect` f S	$\|S\| \log \|S\|$	$\log^2 \|S\|$
`drop` (S, n)	1	1
`empty` ()	1	1
`enum` S	$\|S\|$	1
`filter` f S	$\sum_{s \in S} W(f(s))$	$\log \|S\| + \max_{s \in S} S(f(s))$
`flatten` S	$\|S\| + \sum_{e \in S} \|e\|$	$\log \|S\|$
`fromList` S	$\|S\|$	$\|S\|$
`inject` I S	$\|I\| + \|S\|$	1
`iter` f b_0 S	$O\left(\sum_{i=0}^{\|S\|-1} W(f(b_i, S_i))\right)$	$\sum_{i=0}^{\|S\|-1} S(f(b_i, S_i))$
`iterh` f b_0 S	$O\left(\sum_{i=0}^{\|S\|-1} W(f(b_i, S_i))\right)$	$\sum_{i=0}^{\|S\|-1} S(f(b_i, S_i))$
`length` S	1	1
`map` f S	$\sum_{e \in S} W(f(e))$	$\max_{e \in S} S(f(e))$
`map2` f S_1 S_2	$\sum_{i=0}^{\min(\|S_1\|, \|S_2\|)-1} W(f(S_{1i}, S_{2i}))$	$\max_{i=0}^{\min(\|S_1\|, \|S_2\|)-1} S(f(S_{1i}, S_{2i}))$
`merge` f S_1 S_2	$\|S_1\| + \|S_2\|$	$\log(\|S_1\| + \|S_2\|)$
`nth` S i	1	1
`reduce` f b S	$O\left(\|S\| + \sum_{f(x,y) \in \mathcal{O}_r(f,b,S)} W(f(x,y))\right)$	$\log \|S\| \max_{f(x,y) \in \mathcal{O}_r(f,b,S)} S(f(x,y))$

ArraySequence	*Work*	*Span*
`rev` S	$\lvert S\rvert$	1
`scan` $f\ b\ S$	$\lvert S\rvert + \sum_{f(x,y)\in\mathcal{O}_s(f,b,S)} W(f(x,y))$	$\log\lvert S\rvert \max_{f(x,y)\in\mathcal{O}_s(f,b,S)} S(f(x,y))$
`scani` $f\ b\ S$	$\lvert S\rvert + \sum_{f(x,y)\in\mathcal{O}_s(f,b,S)} W(f(x,y))$	$\log\lvert S\rvert \max_{f(x,y)\in\mathcal{O}_s(f,b,S)} S(f(x,y))$
`showl` S	1	1
`showt` S	1	1
`singleton` e	1	1
`sort` $f\ S$	$\lvert S\rvert \log\lvert S\rvert$	$\log^2\lvert S\rvert$
`subseq`(S,n)	1	1
`tabulate` $f\ n$	$\sum_{i=0}^{n-1} W(f(i))$	$\max_{i=0}^{n-1} S(f(i))$
`take`(S,n)	1	1
`toString` $f\ S$	$\sum_{e\in S} W(f(e))$	$\sum_{e\in S} S(f(e))$
`unzip` S	$\lvert S\rvert$	1
`upmap2` $f\ S$	$\sum_{e\in S} W(f(e))$	$\max_{e\in S} S(f(e))$
`zip` $S_1\ S_2$	$\max(\lvert S_1\rvert, \lvert S_2\rvert)$	1

For `reduce`, $\mathcal{O}_r(f,i,S)$ represents the set of applications of f as defined in the documentation. For `scan`, $\mathcal{O}_s(f,i,S)$ represents the applications of f defined by the implementation of scan in the lecture notes. For `iter` and `iterh`, $b_i = f(b_{i-1}, S_{i-1})$. For `showti`, `argmax`, `merge`, `sort`, `collate`, `collect`, `fields`, and `tokens` the given costs assume that the work and span of the application of f is constant.