In lab exercises

Recall for parallel algorithms, the work law states $T_P \geq T_1/P$ while the span law states $T_P \geq T_\infty$. A greedy scheduler can upper bound T_P at $T_P \leq \frac{T_1 - T_\infty}{P} + T_\infty$.

1. For a fixed input size n, two parallel solutions are developed for a problem of moderate interest. The first program has work $T_1 = 2048$ and span $T_\infty = 1$. The second program has work $T_1 = 1024$ with span $T_\infty = 8$. Assume the runtime for P processors is given by $T_P = T_1/P + T_\infty$.
   (a) Suppose $P \leq 32$ is small. Which program should we use?
   (b) Suppose $P \geq 512$ is large. Which program should we use?
   (c) For what value of P are the run times roughly equal?
2. Suppose a set of experiments run on a greedy scheduler yield the following times: $T_4 = 80, T_{10} = 42, T_{64} = 10$. Using the work and span laws, and the greedy scheduler runtime, argue that this experiment seems flawed. You will need to first find upper bounds on T_1 and T_∞, and then use these bounds to bound T_P.
3. Develop a parallel solution for transposing a matrix which is free of race conditions. Evaluate the work and span of your solution. The transpose A' of a matrix A satisfies $a'_{ij} = a_{ji}$ for $1 \leq i, j \leq n$.
4. Consider the following parallel algorithm for adding two arrays A and B into a third array C.

Algorithm 1 SUM-ARRAY(A, B, C):

```
n = len(A)
blockSize = ...
nblocks = ⌈n/blockSsize⌉
for k = 0 to nblocks − 1:
    spawn ADD-SUBARRAY(A, B, C, k · blockSize + 1, min((k + 1) · blockSize, n))
sync
```

Algorithm 2 ADD-SUBARRAY(A, B, C, i, j):

```
for k = i to j:
    C[k] = A[k] + B[k]
```

   (a) Analyze the parallelism when $blockSize = 1$.
   (b) What is the optimal $blockSize$?
   (c) What is the parallelism if we use parallel loops instead of this blocking strategy?