A Sequential Task Specification

- an example of how hard it is to find a problem with no parallelism
  - taken from…that’s right!…ECE190
  - another ECE190 MP3 (Fall 2008), with some simplifications

- a game for two players (Wythoff’s game)
  - two piles of sticks
  - goal: take the last stick(s)
  - allowed moves
    - take any # > 0 from either pile
    - take the same # > 0 from both piles

- note
  - no ties, no move cycles, finite # steps to goal (bounded by x + y)
  - thus all positions are forced win/forced loss if played correctly

- question
  - which positions are forced win/loss?
  - (lots of wins; we’ll focus on specifying forced loss positions)
• start with the goal state, (0,0)
  – if \((x,y) = (0,0)\) on your turn, you lose
  – thus it’s a forced lose

• first set of forced wins
  – any position from which you can reach the goal state in one move
  – for any \(P > 0\)
    • \((P,0)\) or
    • \((P,P)\) or
    • \((0,P)\)
• any position
  – for which all possible moves are forced wins
  – is a forced loss
  – thus (1,2) and (2,1) are forced losses

• Which induces more forced wins: for any P > 1,
  – (P+1,1) or
  – (P,2) or
  – (P+1,P) or
  – (P,P+1) or
  – (2,P) or
  – (1,P+1)

  the next forced loss is (3,5) and (5,3), and so forth

• but let’s use induction…
starting point
  – for simplicity, write forced loss cases as (x,y) with x < y
  – observe that any forced loss (x,y) differs by some amount; call it k=y-x
    • base case (0,0) has k=0
    • next couple of cases, (1,2) and (3,5), have k=1, k=2
  – forced loss case for any k is unique
    • assume two cases for some k: (x,x+k) and (z,z+k)
    • assume x<z without loss of generality
    • but (z,z+k) can move to (x,x+k) in one move
      (take z-x from both piles)
    • contradiction: (z,z+k) is not a forced loss, but a forced win!
    – however, forced loss case may not exist for all k

now use algorithmic induction
  – base case is (0,0); we know it works and thus k=0 case exists
  – assume sequence (x0,y0), (x1,y1), …, (x(k-1), y(k-1)) to some k
  – find method
    • to determine xk and
    • show that (xk, xk + k) is a forced loss

solution: let xk be
  – the smallest whole number
  – that does not appear in any previous x or y value
  – note: good luck parallelizing the search!

proof
  – three possible move types from (xk, xk + k)
  – we’ll consider one at a time
  – show that all result in forced wins
first two moves: reduce \( x_k \) OR reduce both

- for some \( p > 0 \)
  - \( (x_k, x_k + k) \) moves to \( (x_k - p, x_k + k) \) OR
  - \( (x_k, x_k + k) \) moves to \( (x_k - p, x_k + k - p) \)

- by choice of \( x_k \), \( x_k - p \) appears in a previous forced loss

- thus, for some \( i \), either \( x_i = x_k - p \) or \( x_i + i = x_k - p \)

- first case
  - forced loss at \( (x_k - p, x_k - p + i) \)
  - note that \( i < k \), so \( x_k - p + i < x_k + k - p < x_k + k \)
  - and both \( (x_k - p, x_k + k - p) \) and \( (x_k - p, x_k + k) \) are forced wins

- second case
  - forced loss at \( (x_k - p - i, x_k - p) \)
  - reverse the indices: clearly \( x_k - p - i < x_k + k - p < x_k + k \)
  - again, both \( (x_k - p, x_k + k - p) \) and \( (x_k - p, x_k + k) \) are forced wins
last move: reduce $x_k + k$

- for some $p > 0$, $(x_k, x_k + k)$ moves to $(x_k, x_k + k - p)$

- first case: $|k-p| < k$
  - difference $k - p$ is now covered by some previous forced loss case
  - but $x_k$ does not appear (by choice of $x_k$) in that case
  - thus result is forced win (forced loss for any $k$ is unique)

- second case: $|k-p| \geq k$ (which means $p \geq 2k$)
  - reverse indices to put smaller value first: $(x_k + k - p, x_k)$
  - $x_k + k - p < x_k$, thus $x_k + k - p$ appears in a previous forced loss
  - for some $i$, either $x_i = x_k + k - p$ or $x_i + i = x_k + k - p$
  - first case
    - forced loss at $(x_k + k - p, x_k + k - p + i)$
    - again, since $i < k$, $x_k + k - p + i < x_k$
    - and $(x_k + k - p, x_k)$ is a forced win
  - second case
    - forced loss at $(x_k + k - p - i, x_k + k - p)$
    - reverse the indices: clearly $x_k + k - p - i < x_k$
    - again, $(x_k + k - p, x_k)$ is a forced win

- and…
  - there is a closed-form solution!
  - (didn’t see one last time I looked, but it’s old…maybe as old as 1907?)
  - however, I suggest that you try to find it yourself
  - since without it you can’t parallelize…
  - [darn]