ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

HS 2023

Prof. R. Wattenhofer
Béni Egressy

Computational Thinking
Sample Solutions to Exercise 3

1 Hamiltonian

a) We need to prove that we can solve an arbitrary instance of the Hamiltonian Cycle problem by solving instances of the Hamiltonian Path problem. Let $G = (V, E)$ be the input graph. For each edge $e = (u, v) \in E$, we construct a graph G' as follows. We remove e from G and add two new nodes s and t and two new edges (s, u) and (v, t). We then search G' for a Hamiltonian path. If a Hamiltonian path exists, it must have endpoints s and t, and together with edge e, it gives us a Hamiltonian Cycle in G. On the other hand, if a Hamiltonian cycle exists in G, then for each of its edges, our construction leads to a G' containing a Hamiltonian path.

b) Let $G = (V, E)$ be an input of Hamiltonian Path. For every pair of vertices $u, v \in V$, we construct a graph G' by adding the edge (u, v) to G and check if G' contains a Hamiltonian cycle. If we find one, this gives us a Hamiltonian path in G since at most one of the cycle's edges was added. On the other hand, if a Hamiltonian path exists in G, the G' we obtain by adding the edge between its endpoints contains a Hamilton cycle.

2 Circuit Complexity

a) The inclusion $NC^i \subset AC^i$ for $i \geq 0$ is obvious. To prove $AC^i \subset NC^{i+1}$ for $i \geq 0$, we need to replace gates with a large fan-in since these are not allowed in NC^{i+1}. A gate (AND or OR) with fan-in m can be replaced by a binary tree of the same gate with fan-in 2. This tree will have a depth of $\log m$. Since m is no larger than the size of the circuit, the depth of the circuit increases at most by a factor of $\log m = \log(poly(n)) = \mathcal{O}(\log(n))$ through this transformation.

b) PARITY can be solved using a binary tree of XOR gates. (The XOR gates can be constructed from AND, OR and NOT.) Depending on if we are asking if the number of 1s is even or odd, a NOT gate needs to be placed before the output. Since the depth of such a circuit is $\mathcal{O}(\log n)$, PARITY is in NC^1.

c) In NC^0, all gates have a fan-in of at most 2. That means an output bit in a circuit of depth d can only be connected to at most 2^d input bits. In particular, this number is constant when d is constant. Since the output of PARITY depends on all n input bits, it cannot be computed by a circuit in NC^0.

d) Let $x_n \dots x_1 y_n \dots y_1$ denote the input and $z_{n+1} \dots z_1$ the output. It is well known that two n-bit binary numbers can be summed by successively adding two bits and a possible carry bit from the previous addition. (This is a ripple-carry adder (RCA) consisting of a sequence

of n full adders.) If c_k denotes the carry bit from the k-th addition, this can be written as $z_1 = x_1 \oplus y_1$, $c_1 = x_1 \wedge y_1$,

$$c_k = (c_{k-1} \wedge (x_k \vee y_k)) \vee (x_k \wedge y_k)$$
$$z_k = x_k \oplus y_k \oplus c_k$$

for $k = 2, \ldots, n$, and $z_{n+1} = c_n$.

However, since every carry bit depends on the previous one, this would lead to a circuit of depth $\mathcal{O}(n)$. (In the RCA, each full adder uses the carry bit of the previous full adder as an input.) To find a circuit of constant depth, we would like to compute the carry bits c_k independently of each other. Note that the carry bit at position k is 1 if and only if a carry is generated at some position $i \leq k$ and is carried all the way to position k. This fact can be written as

$$c_k = \bigvee_{1 \leq i \leq k} \left(x_i \wedge y_i \wedge \bigwedge_{i < j \leq k} (x_j \vee y_j) \right).$$

Based on this, we can construct a circuit with constant depth and unbounded fan-in.

e) In binary addition, the first (most significant) bit of the result depends on all input bits. By the same reasoning as in c), this cannot be accomplished in a circuit with constant depth and fan-in of 2.