8 Simulations between Networks

In this section we show how to simulate communication steps of networks by other networks. Suppose that some network H (called *host*) has to simulate some network G (called *guest*). There are three different types of simulations:

- type-1-simulations (simple embedding): every node of G is simulated by exactly one fixed node in H
- type-2-simulations (multiple embedding): every node of G is simulated by at least one fixed node in H
- type-3-simulations (dynamic embedding): every node of G is simulated at any time point by at least one node in H

For any of these simulations we say that H simulates G with *slowdown* s if, for any t, H needs at most $s \cdot t$ communication steps to simulate t communication steps of G.

Below, we will only deal with type-1-simulations in which the networks G and H are (asymptotically) of the same size and every node in H simulates at most one node of G. We note, however, that for certain simulation problems other strategies can be much more efficient. For a survey of various embedding techniques we refer to the paper by Monien and Sudborough.

8.1 Universal networks

First, we will look at how to simulate networks of low degree.

Definition 8.1 *A class $\mathcal{N} = \{G_1, G_2, \ldots\}$ of networks is called* realistic *if there is a constant $c \in \mathbb{N}$ so that all networks in $\mathcal{N}$ have a degree of at most c.*

We start with a simple fact that results from representing each node in some guest graph G with degree d by a binary tree of nodes with d leaves.

Fact 8.2 *Every realistic network of size n can be simulated with constant slowdown by a network of size $O(n)$ and degree 3.*

Next, we ask whether there are networks of small degree that can simulate a large class of realistic networks.

Definition 8.3 *Let $\mathcal{N}$ be a realistic class of networks. A network H is called k-*universal *for $\mathcal{N}$ if every network $G \in \mathcal{N}$ can be simulated with slowdown $O(k)$. If $\mathcal{N}$ represents the class of all realistic networks of size n, then we call H (n,k)-*universal.

To simplify the search for (n,k)-universal networks, we use the following result whose proof can be found in the book by Berge. Recall that a *matching* is a graph in which every node has a degree of at most 1.

Lemma 8.4 *Every graph of degree d can be decomposed into at most $d+1$ matchings.*

This allows us to prove the following result.

Theorem 8.5 *Suppose we have an arbitrary network G of degree d. Then any communication step in G can be simulated by routing at most $d+1$ (partial) permutations.*

Proof. To simplify the arguments, we assume that each node j in G is mapped to node j of the graph simulating G. According to Lemma 8.4, G can be decomposed into at most $d+1$ matchings, say $M_1, \ldots, M_{d+1}$. For each matching M_i we construct a (partial) permutation π_i in a way that for every $\{u, v\} \in M_i$, $\pi_i(u) = v$ and $\pi_i(v) = u$.

Consider now an arbitrary communication step in G. If in this step some node u wants to send a packet to v, then $\{u, v\}$ must be an edge in G and therefore an edge in some matching M_i. Thus, the routing according to π_i will ensure that also the node simulating u can send a message to the node simulating v. $\square$

Using our definition of the routing number and the fact that all realistic networks have a constant degree, the theorem implies the following result.

Corollary 8.6 *Every network with n nodes and routing number R is (n, R)-universal.*

Since the $m \times m$-mesh has a routing number of $O(m)$, it is $(n, \sqrt{n})$-universal, where $n = m^2$. The following result shows that the hypercube of size n is even $(n, \log n)$-universal.

Theorem 8.7 *The hypercube of size n has a routing number of $O(\log n)$.*

Proof. Let $n = 2^d$ for some $d \in \mathbb{N}$. Using the standard bit adjustment path system, we know that every permutation can be routed with a dilation of at most $d = \log n$. Furthermore, we know from Section 2 that when using Valiant's trick (first route the packets to random intermediate destinations before routing them to their correct destinations), the expected congestion at an edge is at most 1. However, this does not suffice to bound the routing number, since there could be large differences in the congestion among the edges. Fortunately, it can be shown that this is not possible.

Suppose that we want to route some permutation π. Consider some fixed edge e. We know from the proof of Theorem 2.4 that at most $2^{d-1} = n/2$ paths pass through e. Let P_e be the set of all these paths. Furthermore, let C_e^1 be the number of paths in P_e that are selected in the first part of Valiant's trick, and let C_e^2 be the number of paths in P_e that are selected in the second part of Valiant's trick. First, we bound C_e^1. We know that for any path $p \in P_e$ the probability that p is selected is equal to $1/n$. Furthermore, paths from different sources have independent probabilities, and no two paths with the same source can be selected simultaneously. Hence, for $c \geq \log n$ we get

$$\begin{aligned}
\Pr[C_e^1 \geq c] &= \Pr[\bigcup_{p_1,\ldots,p_c \in P_e} (\text{paths } p_1, \ldots, p_c \text{ are selected})] \\
&\leq \sum_{p_1,\ldots,p_c \in P_e} \Pr[\text{paths } p_1, \ldots, p_c \text{ are selected}] \\
&\leq \binom{n/2}{c} \left(\frac{1}{n}\right)^c \\
&\leq \frac{(n/2)^c}{c!} \cdot \frac{1}{n^c} = \frac{1}{2^c \cdot c!} \leq \frac{1}{2^{3c+1}} = \frac{1}{2n^3}
\end{aligned}$$

if c is sufficiently large. The same probability bound can be shown for C_e^2. Hence, the probability that for the total congestion at e, $C_e = C_e^1 + C_e^2$, it holds $C_e \geq 2\log n$ is at most $1/n^3$. Since the hypercube has at most $n \log n$ edges, it follows that the probability that there exists an edge e with $C_e \geq 2\log n$ is at most $1/n$. Thus, there exists a choice (which will even happen with high probability) for the random intermediate destinations so that for all edges e, $C_e \leq 2\log n$.

To summarize, we showed that for every permutation there is a collection of paths with dilation and congestion at most $2\log n$. Due to the fact that for any (simple) collection of paths with dilation D and congestion C, packets can be sent along this collection in time $O(C+D)$ (see Assignment 4), we conclude that the hypercube has a routing number of $O(\log n)$. □

Similar results can also be shown for the deBruijn graph (see the assignment). Is it possible to have a constant degree graph of size $O(n)$ that is $(n, o(\log n))$-universal? The following theorem due to Meier auf der Heide, Storch and Wanka shows that such a graph does not exist.

Theorem 8.8 *For any constant degree (n,k)-universal network of size m, $m \cdot k = \Omega(n \cdot \log n)$.*

8.2 Simulation of networks of arbitrary degree

Now we look at the problem of simulating networks of arbitrary degree. To simplify this problem, we first assume that even though the degree of a node may be very high, the communication pattern in one time step always has to ensure that it is a (partial) permutation. (We will present a more general model in the assignment.)

Definition 8.9 *A network of size n is a (n,k)-*simulator *if it can simulate every network of size n with slowdown $O(k)$.*

Theorem 8.10 *Let H be any network that can sort n numbers in time $t(n)$. Then H is a $(n, t(n))$-simulator.*

Proof. Let G be the network H wants to simulate. Since every node in G can send and receive at most one packet, the simulation of one communication step in G involves routing a partial permutation in H. If the permutation is complete, we can directly use sorting. Otherwise, we use the following strategy.

Let the nodes in H be numbered from 0 to $n-1$. To simplify the description, assume that node i of G is mapped to node i of H. All nodes in H whose node in G intends to send a packet, generate a packet with the corresponding destination. All other nodes in H generate a so-called *ghost packet* with destination n. If we now sort the packets according to their destinations, then the nodes of highest numbers will be occupied by the ghost packets.

Next, we route a permutation in which each node i sends the destination of the packet received by it to node $i-1$. For this we can again use the sorting algorithm. This allows to find out which node stores the non-ghost packet with maximum destination number. Suppose that the number of this node is m. This number m is afterwards broadcasted to all other nodes. (Simply use the minimum spanning tree for this. The time required to send m along all edges of the spanning tree is bounded by the diameter of H, which is clearly below $t(n)$.)

Now, the nodes do the following. Each node i with a ghost packet assigns this packet a destination number of $(i-m)n$. Each node i with a non-ghost packet with destination j assigns this packet a destination of $(j-i)n+j$.

Next, we again sort all packets according to their destinations. It is not difficult to verify that now all non-ghost packets reach their correct destination. □

Since we know from the previous assignment that in a deBruijn graph of size n every permutation can be routed in $O((\log n)^2)$ steps, the theorem implies that the deBruijn graph is a $(n, (\log n)^2)$-simulator. Next we look at lower bounds. For this the following result will be helpful.

Lemma 8.11 *If H is a $(n, t(n))$-simulator, then the routing number of H is bounded by $O(t(n))$.*

Since every realistic network of size n must have a routing number of $\Omega(\log n)$, the lemma implies the following result.

Theorem 8.12 *There can be no realistic network that is a $(n, o(\log n))$-simulator.*

Hence, the best one can hope for is to find a realistic network that is a $(n, \log n)$-simulator. Such networks indeed exist: using the AKS sorting network mentioned in the previous lecture, a network can be constructed that is a $(n, \log n)$-simulator.

8.3 References

- C. Berge. *Graphs and Hypergraphs.* North-Holland Publishing Company, Amsterdam, 1976.
- F. Meyer auf der Heide, M. Storch, and R. Wanka. Optimal trade-offs between size and slowdown for universal parallel networks. In *Proceedings of the 7th ACM Symposium on Parallel Algorithms and Architectures*, 1995.
- B. Monien and H. Sudborough. Embedding one interconnection network into another. *Computing*, 7:257–282, 1990.