# 2 Oblivious Routing

We call a routing strategy *oblivious* if the path a packet takes only depends on its source and destination. In other words, an oblivious routing strategy is based on a *fixed* collection of paths, one path for each source-destination pair. Such a path collection will be called a *path system* in the following. In this case, every routing problem for a collection of source-destination pairs reduces to a switching problem for a collection of paths. We will use two important parameters to study the time required to send packets along a fixed collection of paths:

- the *congestion* $C$, which represents the maximum number of paths that share an edge, and
- the *dilation* $D$, which represents the maximum length of a path (measured in the number of edges).

Obviously, the following fact holds.

**Fact 2.1** *For any path collection with congestion $C$ and dilation $D$, at least* $\max[C, D]$ *steps are required to send packets along these paths, one packet per path.*

Thus, in order for an oblivious routing strategy to be efficient, an important prerequisite is that the congestion and dilation are small. We will concentrate in this section on how well oblivious routing strategies are able to keep these parameters small.

## 2.1 The Borodin-Hopcroft Lower Bound

The following theorem demonstrates that no matter what kind of oblivious strategy is used, there is always a permutation that causes a high congestion.

**Theorem 2.2 (Borodin, Hopcroft)** *For every graph $G$ of size $n$ and degree $d$ and every oblivious routing strategy, there is a permutation $\pi$ that requires a path collection of congestion $C_\pi \geq \sqrt{n/d}$ at some node.*

**Proof.** Let $[n] = \{0, \ldots, n-1\}$ represent the set of all nodes in $G$ and let $\mathcal{A}$ be any oblivious routing algorithm for $G$. Furthermore, let $\mathcal{P} = \{p_{i,j} : \ i, j \in [n]\}$ be the path system induced by $\mathcal{A}$. A node $s$ is called a *source* for node $i$ w.r.t. $t$ if $p_{s,t}$ moves through $i$. In Figure 1, for example, $s_3$ is a source for $i$ w.r.t. $t$.

In the following, we will construct a permutation $\pi$ with a high congestion. First we show that for any node $t$ there are many nodes that have many sources w.r.t. $t$. Let $A(t, z) = \{i \in [n] : \ i$ has w.r.t. $t$ at least $z$ sources$\}$ be the set of all nodes that are contained in at least $z$ different paths of $\mathcal{P}$ that lead to $t$. Then the following lemma holds.

**Lemma 2.3** *For all $t \in [n]$: $|A(t, z)| \geq \frac{n}{d \cdot z}$.*

**Proof.** For any fixed $t \in [n]$, let $L = \{p_{s,t} : \ s \in [n]$ and $s \notin A(t, z)\}$, or in words, the number of paths, that start outside of $A(t, z)$, and let $B \subseteq L$ be the set of all direct neighbors of nodes in $A(t, z)$ that are not in $A(t, z)$.

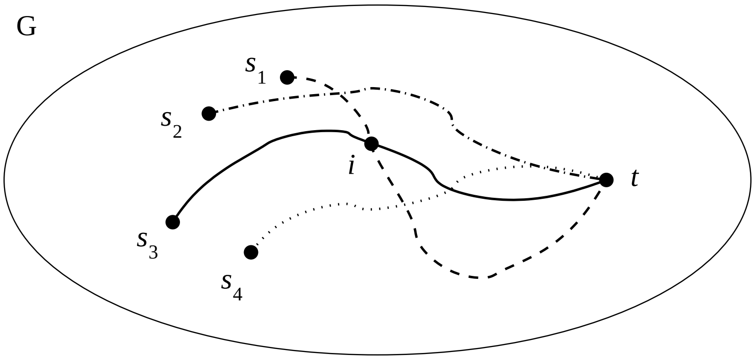


Figure 1: Illustration of the paths to $k$.

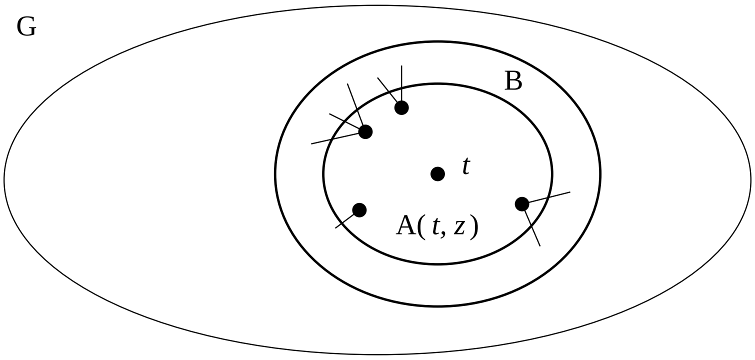


Figure 2: Illustration of $A(t,z)$ and $B$.

Obviously, $|L| = n - |A(t,z)|$. Since the maximum degree of $G$ is $d$, it further holds that $|B| \leq |A(t,z)| \cdot d$. Because $B \cap A(t,z) = \emptyset$, every node in $B$ has at most $z-1$ paths that lead to $t$. Hence, $|B| \cdot (z-1) \geq L$ and therefore

$$\begin{aligned} & |A(t,z)| \cdot d \cdot (z-1) \geq |B| \cdot (z-1) \geq |L| \\ \Rightarrow \quad & |A(t,z)| \cdot d \cdot (z-1) \geq n - |A(t,z)| \\ \Rightarrow \quad & |A(t,z)| \cdot (d \cdot (z-1) + 1) \geq n \\ \Rightarrow \quad & |A(t,z)| \geq \frac{n}{d \cdot (z-1) + 1} \geq \frac{n}{d \cdot z} \,. \end{aligned}$$

$\sqcap$

Now, let $X(z) = \{(i,t) :\ i,t \in [n] \text{ and } i \in A(t,z)\} = \bigcup_{t\in[n]}(A(t,z) \times \{t\})$. Then it holds

$$|X(z)| = \sum_{t\in[n]} |A(t,z)| \overset{\text{Lemma 2.3}}{\geq} n \cdot \frac{n}{d \cdot z} = \frac{n^2}{d \cdot z} \,.$$

For every node $i$ let $T_i = \{t : (i,t) \in X(z)\}$ be the set of all destinations for which at least $z$ paths move through $i$. Since

$$\sum_{i \in [n]} |T_i| = |X(z)| \geq \frac{n^2}{d \cdot z}$$

but on the other hand there are only $n$ sets $T_i$, there must exist a node $i$ with $|T_i| \geq \frac{n}{d \cdot z}$. Choose $z$ so that $z$ and $|T_i|$ are of the same size. This is the case for $z = \frac{n}{d \cdot z}$ or $z = \sqrt{n/d}$.

Thus, there must be a node $i$ for which there are at least $\sqrt{n/d}$ destinations that have at least $\sqrt{n/d}$ paths through $j$. Simply choosing for all of these destinations one after the other any source that has not been chosen by a previous destination results in a partial permutation with congestion at least $\sqrt{n/d}$. ⊓

## 2.2 Valiant's Trick

We saw in Section 2.1 that oblivious algorithms can have an extremely high congestion. A solution discovered by Valiant is able to significantly reduce the congestion with high probability.

**Valiant's Trick:**
For every source-destination pair $(s,t)$, choose a random intermediate destination $v \in [n]$ and send the packet first along the path $p_{s,v}$ and then along the path $p_{v,t}$.

To demonstrate the effect of this trick, let us consider routing in the $d$-dimensional hypercube. Suppose that we use for every source-destination pair $(s_{d-1}, \ldots, s_0), (t_{d-1}, \ldots, t_0) \in \{0,1\}^d$, the path that first adjusts $s_0$ to $t_0$, then $s_1$ to $t_1$, and so on, until all bits have been set to the destination's values. Using Theorem 2.2, there must be a permutation that has a congestion of at least $\sqrt{2^d/d}$. However, if we use Valiant's trick, we arrive at the following result.

**Theorem 2.4** *Using Valiant's trick in the d-dimensional hypercube, any permutation is routed along paths with dilation at most* $2d$ *and an expected congestion of 1 in any direction of any edge.*

**Proof.** Our oblivious path selection strategy ensures that every path in the path system has a length of at most $d$. Thus, Valiant's trick produces paths of length at most $2d$. Now, consider some fixed directed edge $e$ in dimension $i \in \{1, \ldots, d\}$. Our path selection strategy implies that there are $2^{i-1}$ source nodes that can reach $e$ and $2^{d-i}$ destination nodes that can be reached via $e$. Thus, altogether $2^{i-1+d-i} = 2^{d-1}$ paths move through $e$. When using Valiant's trick, any permutation routing problem decomposes into two problems in which every node chooses one random destination (one from the view of the sources and one from the view of the destinations, see Figure 3).

We refer to the first of these problems as part 1 and the second as part 2. Since there are $2^d$ random intermediate destinations to choose from, we obtain for every pair of nodes $(s,i)$ in part 1 that

$$\Pr[\text{path } p_{s,i} \text{ is chosen}] = \frac{1}{2^d} .$$

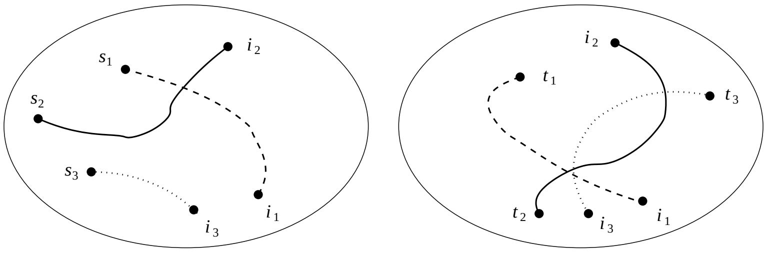


Figure 3: Illustration of Valiant's trick.

Similarly, it holds for pair of nodes $(i, t)$ in part 2 that

$$\Pr[\text{path } p_{i,t} \text{ is chosen}] = \frac{1}{2^d} .$$

Consider now some fixed edge $e$ of the hypercube, and let $P_e$ be the set of all pairs of nodes $(i, j)$ whose path $p_{i,j}$ traverses $e$. For all $(i, j) \in P_e$, let the binary random variable $X_{i,j} \in \{0, 1\}$ be 1 if and only if $p_{i,j}$ was chosen in part 1, and let the binary random variable $Y_{i,j} \in \{0, 1\}$ be 1 if and only if $p_{i,j}$ was chosen in part 2. Then the random variable

$$X = \sum_{(i,j)\in P_e} (X_{i,j} + Y_{i,j})$$

represents the number of all selected paths that traverse $e$ when using Valiant's Trick. We want to bound the expected number of these paths. For every $(i, j) \in P_e$, the expected value of $X_{i,j}$ is

$$\mathrm{E}[X_{i,j}] = \sum_{x\in\{0,1\}} x \cdot \Pr[X_{i,j} = x] = \Pr[X_{i,j} = 1] = \frac{1}{2^d} .$$

The same holds for $Y_{i,j}$. Due to the linearity of expectation, we therefore get

$$\mathrm{E}[X] = \mathrm{E}\left[\sum_{(i,j)\in P_e} (X_{i,j} + Y_{i,j})\right] = \sum_{(i,j)\in P_e} (\mathrm{E}[X_{i,j}] + \mathrm{E}[Y_{i,j}]) = 2^{d-1} \cdot 2 \cdot \frac{1}{2^d} = 1 .$$

$\sqcap$

## 2.3 References


- A. Borodin and J.E. Hopcroft. Routing, merging, and sorting on parallel models of computation. *Journal of Computer and System Sciences*, 30:130–145, 1985.
- L.G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 2(11):350–361, 1982.