


The one-to-many node-disjoint paths problem in certain interconnection networks

Iain A. Stewart¹

Department of Computer Science
Durham University, U.K.

May 2009

¹Joint work with Yonghong Xiang, Durham University. 

Parallel computers

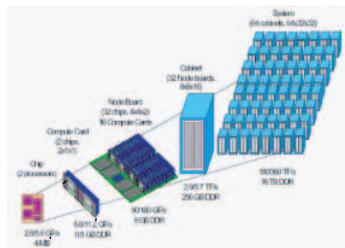
Parallel computers are usually

- ▶ shared-memory machines, or
- ▶ **distributed-memory machines.**

The dominant factor inhibiting faster global computations is **inter-processor communication**:

- ▶ routing mechanisms (how paths are determined)
- ▶ flow control (how buffers/channels are assigned to packets)
- ▶ switching (how a packet is moved)
- ▶ **network topology.**

IBM Blue Gene



Some design parameters

There are often conflicting demands on an interconnection network:

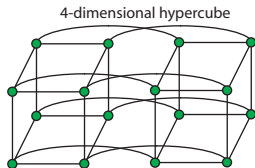
- ▶ symmetry (to aid programming and analysis)
- ▶ small diameter (to reduce message latency)
- ▶ recursive decomposability (to aid scalability)
- ▶ low degree (to reduce communication overheads)
- ▶ **support rapid, easy and efficient inter-processor communication**
- ▶ support the simulation of other machines
- ▶ ...

There is no one network to optimise all parameters and trade-offs have to be made.

Some popular topologies

The n -dimensional hypercube Q_n

- ▶ Vertex set consists of $\{0, 1\}^n$.
- ▶ There is an edge (\mathbf{u}, \mathbf{v}) iff \mathbf{u} and \mathbf{v} differ in exactly one bit.



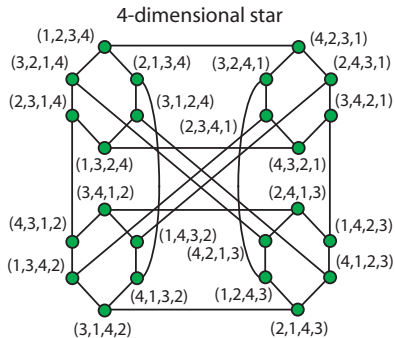
The hypercube Q_n has a number of desirable properties:

- ▶ it is a Cayley graph and so node-transitive
- ▶ it is edge-transitive
- ▶ it has diameter n
- ▶ it is recursively decomposable
- ▶ it has a trivial node-to-node routing algorithm
- ▶ it can efficiently simulate many other networks, e.g., trees, meshes, hexagonal graphs, ...

Some popular topologies

The n -star S_n [Akers, Harel, Krishnamurthy, *Proc. ICPP 87*]

- ▶ Vertex set consists of $\{(v_1, v_2, \dots, v_n) : v_i, v_j = 1, 2, \dots, n, \text{ and } v_i \neq v_j, \text{ for } i \neq j\}$.
- ▶ There is an edge joining (u_1, u_2, \dots, u_n) and (v_1, v_2, \dots, v_n) if $u_1 = v_i$ and $v_1 = u_i$, for some $i \in \{2, 3, \dots, n\}$, with $u_j = v_j$, for $j \in \{2, 3, \dots, n\} \setminus \{i\}$.



Some properties of S_n

The n -star S_n has a number of desirable properties too (in the context of parallel computation):

- ▶ it has degree and diameter $O(\log(n)/\log\log(n))$
- ▶ it is recursively decomposable
- ▶ it is a Cayley graph and so node-transitive
- ▶ it is tolerant of faulty nodes and links in a number of scenarios.

In general, star graphs compare favourably with hypercubes.

(n, k) -stars

A drawback of n -stars is that the ‘gap’ between the sizes of S_n and S_{n+1} (that is, $n!$ and $(n+1)!$) is considerable.

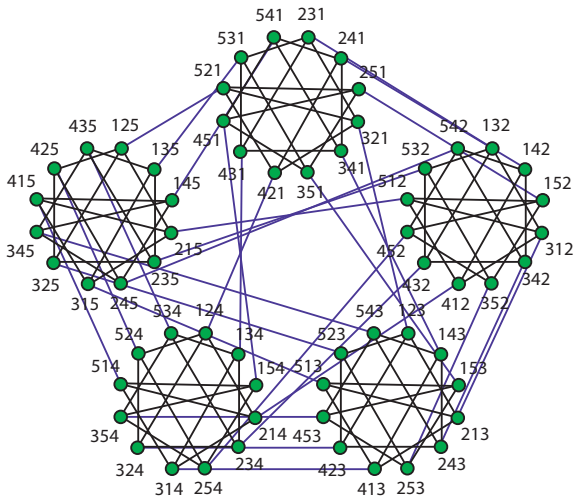
The (n, k) -star $S_{n,k}$ is defined as follows [Chiang, Chen, IPL 95].

- ▶ The node set is $\{(u_1, u_2, \dots, u_k) : \text{each } u_i \in \{1, 2, \dots, n\} \text{ with } u_i \neq u_j, \text{ for } i \neq j\}$.
- ▶ There is an edge $((u_1, u_2, \dots, u_k), (v_1, v_2, \dots, v_k))$ if:
 - ▶ $u_1 \neq v_1$ and $u_j = v_j$, for all $j = 2, 3, \dots, k$ (a 1-edge); or
 - ▶ $u_1 = v_i$ and $v_1 = u_i$, for some i , with $u_j = v_j$, for all $j \in \{2, 3, \dots, k\} \setminus \{i\}$ (an i -edge).

So, $S_{n,k}$ has $\frac{n!}{(n-k)!}$ nodes, $\frac{(n-1)}{2} \times \frac{n!}{(n-k)!}$ edges, and is regular of degree $n-1$.

In particular, $S_{n,n-1}$ is isomorphic to the star S_n , and $S_{n,1}$ is a clique on n nodes.

A (5, 3)-star



(5,3)-star

Some properties of (n, k)-stars

- ▶ They can be recursively decomposed in a number of ways [Chiang, Chen, *IPL* 95].
- ▶ They are node-symmetric [Chiang, Chen, *IPL* 95].
- ▶ They have a simple shortest path routing algorithm [Chiang, Chen, *IJFCS* 98].
- ▶ $S_{n,k}$ has connectivity $n - 1$ [Chiang, Chen, *IJFCS* 98].
- ▶ They have reasonably good embedding and connectivity properties, even in the presence of faults [Chang, Kim, , *Proc. ICPADS* 01], [Hsu, Hsieh, Tan, Hsu, *Networks* 03], [Hsu, Lin, Hung, Hsu, *IJFCS* 06].
- ▶ The wide-diameter of $S_{n,k}$ is either $\Delta(S_{n,k}) + 1$ or $\Delta(S_{n,k}) + 2$, depending upon k and n . [Chiang, Chen, *IJFCS* 98], [Lin, Duh, Cheng, *Proc. CCT* 04], [Lin, Duh, *Inf. Sci.* 08].

Our problem

The **one-to-many node-disjoint paths problem** for $S_{n,k}$ is:

- ▶ given: the graph $S_{n,k}$; a source node u ; and some distinct target nodes v_1, v_2, \dots, v_{n-1}
- ▶ find: $n - 1$ node-disjoint paths from the source to the targets so that the path-lengths are minimized.

Notes:

- ▶ an instance to the problem has size $O(kn \log n)$, even though $S_{n,k}$ has exponentially many (in n) nodes
- ▶ the criterion regarding path-length is satisfied if the length of the longest path is as small as possible
- ▶ we cannot cater for more target nodes as $S_{n,k}$ has degree and connectivity $n - 1$
- ▶ Menger's Theorem tells us that our paths exist but tells us nothing about their lengths.

Disjoint paths in other networks

- ▶ In 1989, Rabin [Rabin, *JACM* 89] showed that given a source node and n distinct target nodes in an n -dimensional hypercube Q_n , there is an $O(n^2)$ time algorithm that builds node-disjoint paths from the source to the targets such that each path has length at most the diameter of Q_n plus 1 (this is optimal).
- ▶ In 1997, Gu and Peng [Gu, Peng, *IPL* 97] showed that given a source node and $n - 1$ distinct target nodes in an n -star graph S_n , there is an $O(n^2)$ time algorithm that builds node-disjoint paths from the source to the targets such that each path has length at most the diameter of S_n plus 2.
 - ▶ It is open as to whether this is optimal as the only lower bound known is the diameter of S_n plus 1.

Recursive decomposability

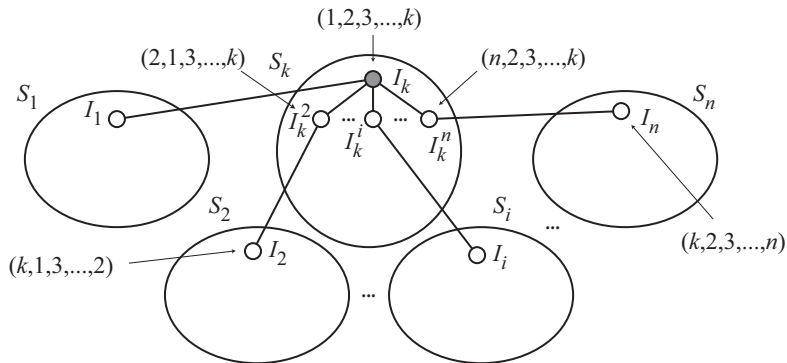
- ▶ If we fix any component of the nodes of $S_{n,k}$, apart from the first, at some $w \in \{1, 2, \dots, n\}$:

$$(v_1, v_2, \dots, v_{i-1}, w, v_{i+1}, \dots, v_k)$$

then we get n disjoint copies of $S_{n-1,k-1}$ (joined by additional edges).

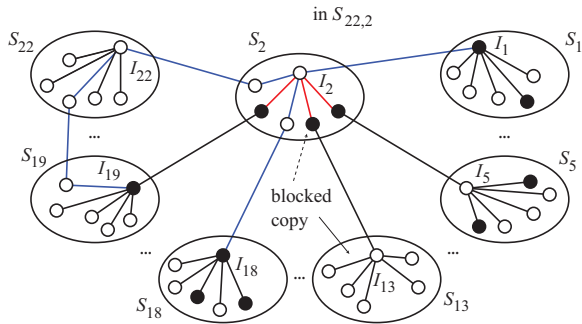
- ▶ Thus, there is scope for us to partition $S_{n,k}$ over some **dimension**.
- ▶ In fact, there is always a dimension over which we can partition $S_{n,k}$ so that we obtain n disjoint copies of $S_{n-1,k-1}$ so that no more than $n - 2$ target nodes lie in any of these copies (thus setting up a recursive algorithm).
- ▶ Further, because $S_{n,k}$ is node-symmetric then we may always assume that our source node is $l_k = (1, 2, \dots, k)$.

The basic set-up



Here, we assume that we partition over dimension k .

The case for $k = 2$



- ▶ Paths in S_2 are reserved first (marked in red).
- ▶ Other paths from targets go 'directly' to I_k or via a copy S_i containing no target nodes and which is not 'blocked'.
- ▶ A 'counting/structural' argument is used to verify that the basic construction is always possible.

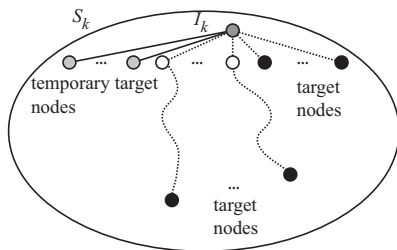
The result for $k = 2$

Theorem

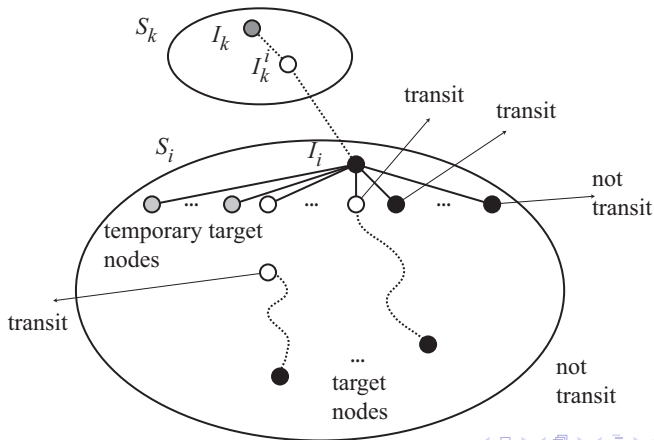
When T is a set of $n - 1$ distinct nodes in $S_{n,2}$ and when l is a node not in T , there is an algorithm that finds $n - 1$ node-disjoint paths from the nodes in T to the node l . Furthermore, all paths found have length at most 5, which is optimal, and the time complexity of the algorithm is $O(n^2)$.

The general case

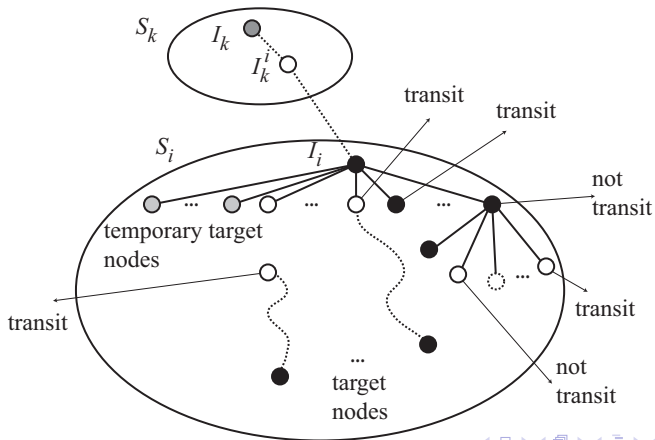
- ▶ First, we (recursively) find node-disjoint paths from the target nodes in S_k to I_k (if any such target nodes exist); these paths lie wholly within S_k and are not changed throughout the subsequent execution of the algorithm.
- ▶ We set neighbours of I_k to be temporary target nodes to keep as much scope as possible for building other paths through sets of transit nodes and on to I_k .



- ▶ Next, we work through the other S_i 's in turn and via recursive calls establish paths from the target nodes in each S_i to I_i . We then amend and extend these paths so that they ultimately lead to I_k .



- Next, we work through the other S_i 's in turn and via recursive calls establish paths from the target nodes in each S_i to I_i . We then amend and extend these paths so that they ultimately lead to I_k .



The resulting algorithm

What results is a recursive algorithm on $S_{n,k}$ that can be shown to

- ▶ make at most $(k - 1)(n - 2)^2$ recursive calls
- ▶ with each recursive call taking $O(k^2 n^2)$ time
- ▶ so that the longest path constructed from the source to a target node is $6k - 7$ (solving $p_k \leq p_{k-1} + 6$ and $p_2 = 5$).

Theorem

When T is a set of $n - 1$ distinct nodes in $S_{n,k}$ and when l is a node not in T , there is an algorithm that finds $n - 1$ node-disjoint paths from the nodes in T to the node l .

Furthermore, all paths found have length at most $6k - 7$ and the time complexity of the algorithm is $O(k^3 n^4)$.

A comparison with Q_n and S_n

- ▶ Rabin's $O(n^2)$ time algorithm for finding n node-disjoint paths from a source to n targets in Q_n with the longest of length at most 1 plus the diameter
- ▶ Gu and Peng's $O(n^2)$ time algorithm for finding $n - 1$ node-disjoint paths from a source to $n - 1$ targets in S_n with the longest of length at most 2 plus the diameter.
- ▶ Our algorithm takes longer to run and the length of the longest path is at most roughly 3 times the diameter.
- ▶ In order to obtain an improvement using our techniques, we would need to increase the length of our paths by at most 2 at each recursive call and this seems extremely unlikely/difficult.
- ▶ Of course, the more complex structure of $S_{n,k}$ could very well mean that our result is, in fact, near optimal.