# Abstract

**As transistor sizes shrink and we approach the ``end of Moore's law'', interconnects, both on-chip and off-chip, will represent the biggest bottleneck for embedded systems designers. Several groups are researching optical interconnects to cope with this trend. Optical interconnects enable new system architectures. These new architectures in turn require new methods for high-level application mapping and hardware/software co-design. In this presentation, we discuss high-level scheduling and interconnect topology synthesis techniques for embedded multiprocessors. We focus on designs that are streamlined for one or more digital signal processing (DSP) applications. That is, we seek to synthesize an *application-specific interconnect topology* for a multiprocessor DSP design. We show that flexible interconnect topologies that allow single-hop communication between processors offer advantages for reduced power and latency.**

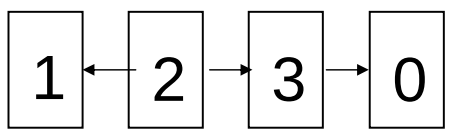
**We have previously shown that multiprocessor scheduling algorithms can deadlock in the general case of a topology graph that is not strongly connected, or if communication is limited to be single hop. We have also demonstrated an efficient algorithm that can be used in conjunction with existing scheduling algorithms for avoiding this deadlock. In this presentation we discuss the advantages of performing application scheduling and interconnect synthesis jointly, and present a probabilistic scheduling/interconnect algorithm utilizing graph isomorphism to pare the design space. We demonstrate the performance advantages that an application-specific interconnect topology can produce for several DSP benchmarks.**

# Deadlock and Flexibility

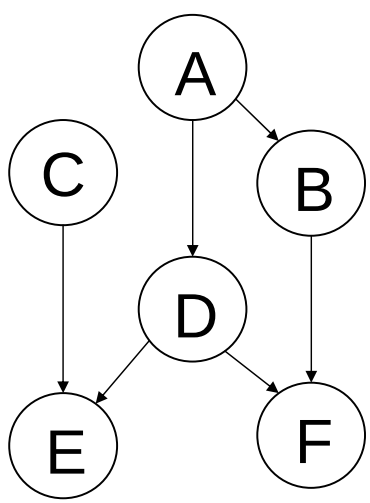
- **Existing scheduling algorithms assume every pair of processors can communicate**
- **Scheduling not well studied for irregular interconnection networks**
- **Can *deadlock* for arbitrary topologies**
  - **Developed algorithms to adapt existing list scheduling algorithms to avoid deadlock**
- **Some scheduling moves have greater *flexibility***

**Partial Schedule**
**A on proc. 2**
**B on proc. 1**

**Constraint Sets**
**F[A] = {2}**
**F[B] = {1}**
**F[F] = {1}**
**F[D] = {1,2}**
**F[E] = {1,2,3}**
**F[C] = {1,2,3}**
**Flexibility = 11/24**




Topology graph

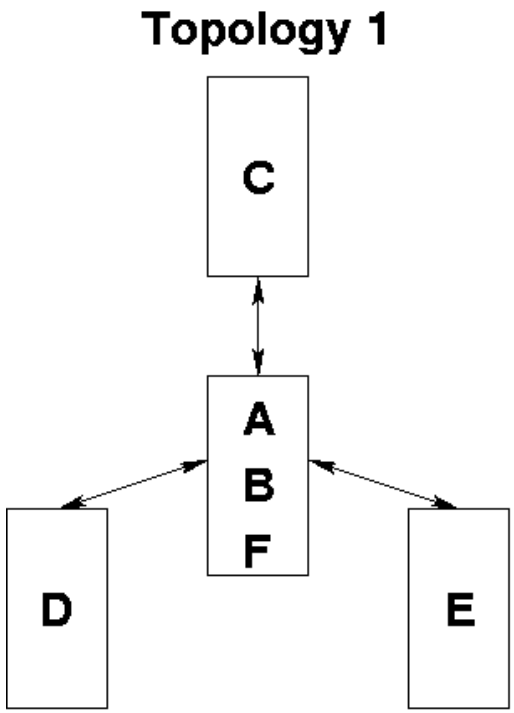


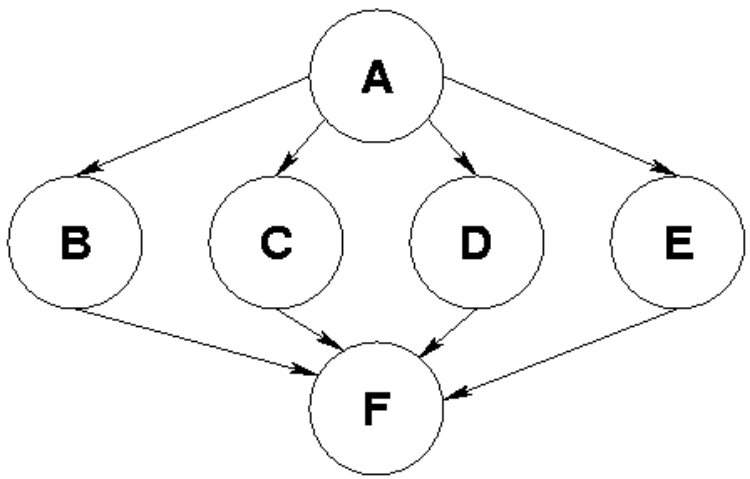

Application graph

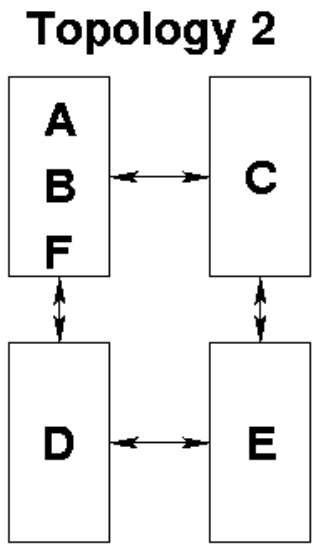
**Partial Schedule**
**A on proc. 2**
**B on proc. 3**

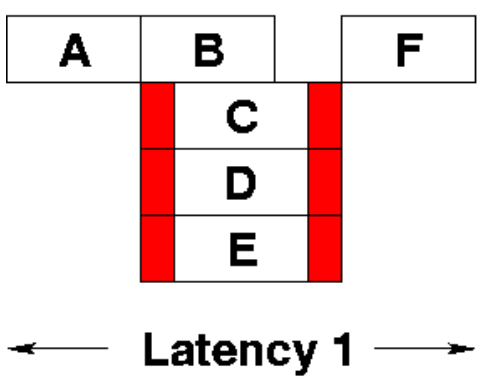
**Constraint Sets**
**F[A] = {2}**
**F[B] = {3}**
**F[F] = {0,3}**
**F[D] = {1,2,3}**
**F[E] = {0,1,2,3}**
**F[C] = {0,1,2,3}**
**Flexibility = 15/24**

# Effect of Topology

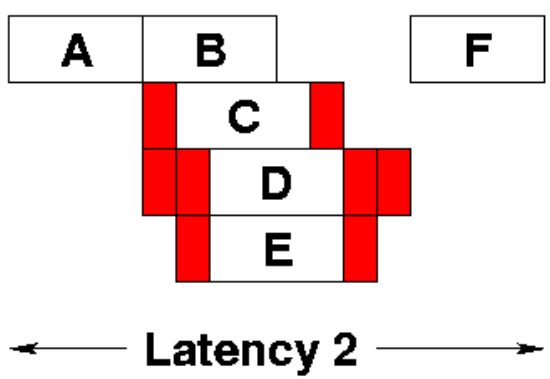


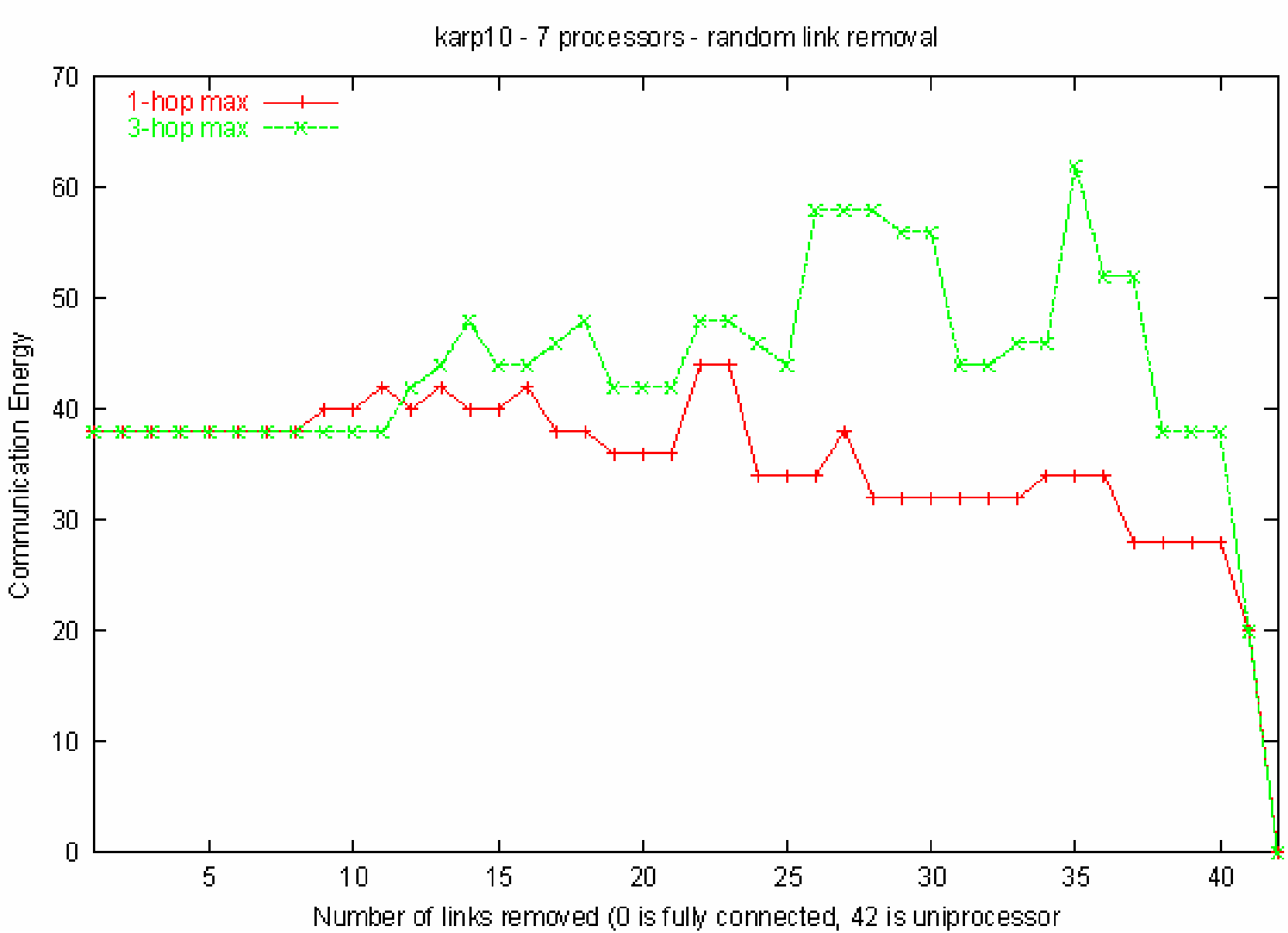




Lower latency
and communication
energy for topology 1



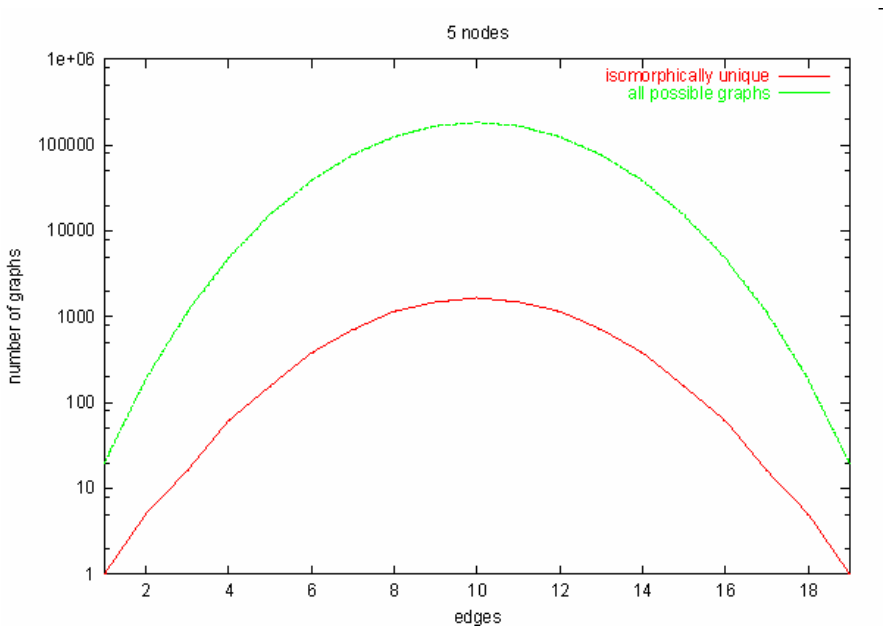
# Low Hop Communication Saves Energy



# Link Synthesis Algorithm

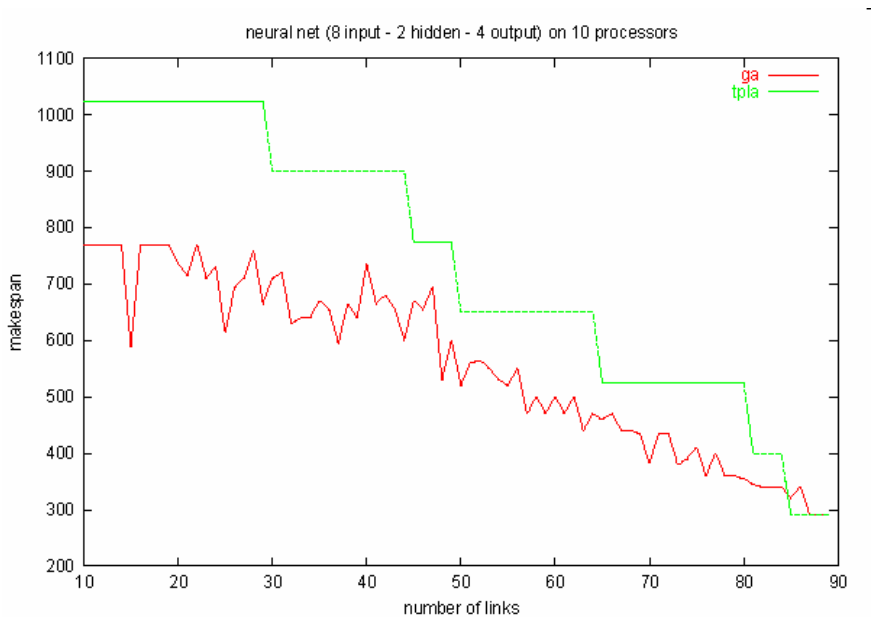
- Developed both deterministic and evolutionary (GA) algorithms
- GA objective utilizes DLS scheduling modified with flexibility metric
  - Crossover operators allow fan-out constraints to be preserved
  - Use graph isomorphism to pare the design space

## Paring the design space




- Consider only isomorphically unique graphs
- Reduction by orders of magnitude

## Link synthesis results




GA (red) outperforms deterministic over a range of topologies