Università della Svizzera italiana

Advanced Learning and Research Institute ALaRI

Online Task Remapping Strategies for Fault-tolerant Network-on-Chip Multiprocessors

Onur Derin, Deniz Kabakcı, Leandro Fiorin

ALaRI
Faculty of Informatics
University of Lugano
Lugano, Switzerland

derino@alari.ch

NOCS'11 - Pittsburgh
May 3, 2011

Outline

Problem

ILP formulation of the optimal mapping problem of KPN applications onto NoC

- Minimization of the communication cost
- Minimization of the total computation time

Online Task Remapping

- Optimal Task Remapping
- Center of Gravity method (CoG)
- Nonidentical Multiprocessor Scheduling Heuristics (NMS)
- Localized NMS Heuristic (LNMS)
- Case study
- Results

Introduction

- Continuity of service support in the MADNESS NoC platform
  - starting point
    - Kahn Process Networks as the computation model
    - xpipes-based NoC from Uni. Cagliari, NORMA model, message-passing support
  - Main tasks
    - a middleware to execute KPN on NoCs
    - fault detection/masking via self-testing and self-checking
    - reconfiguration via **online task remapping**
    - task migration from faulty nodes
    - fault-tolerant interconnect

Problem

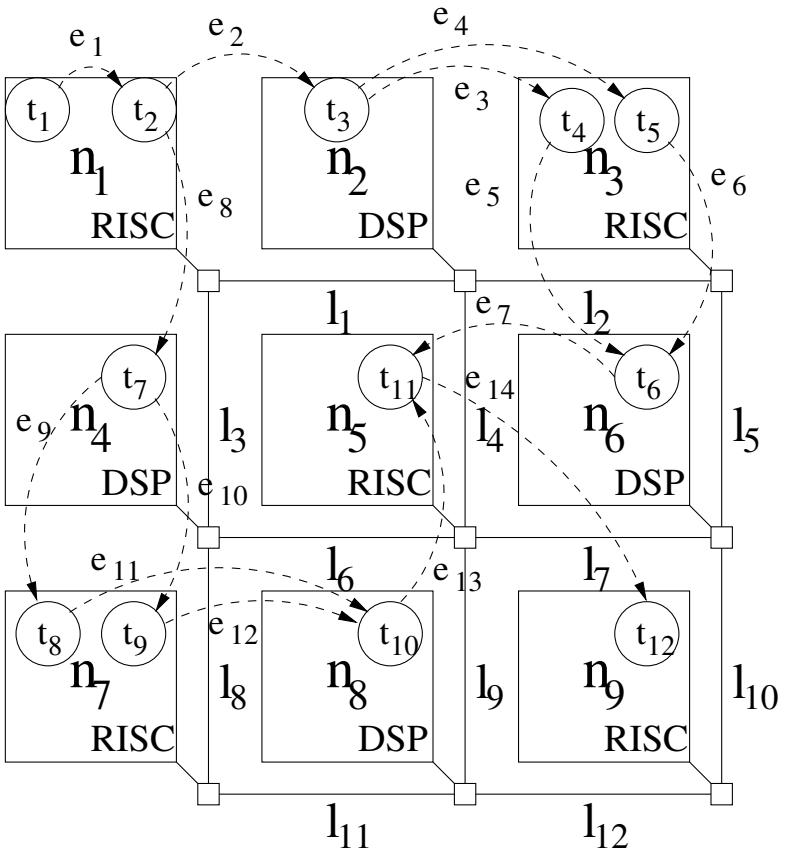


Figure: A KPN application running on a 3x3 mesh

Problem: Where should the tasks on the faulty core be moved?

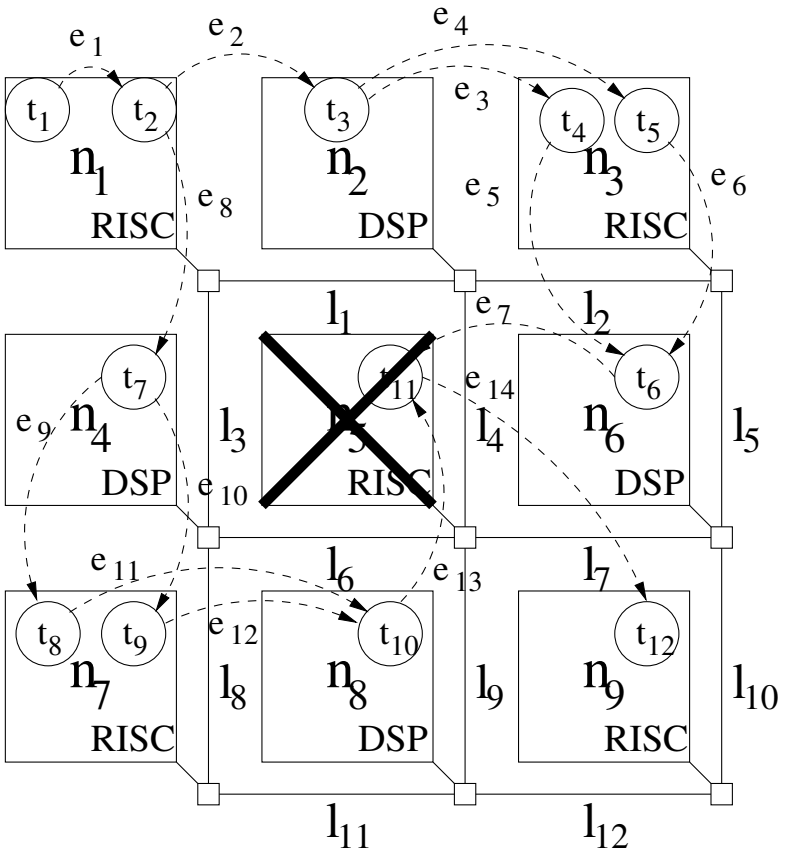


Figure: Processing node n_5 becomes faulty

Approach

- Solve the task mapping problem onto NoC-based heterogeneous multiprocessors **optimally**
- Consider different fault scenarios and find new optimal remappings
- Propose heuristics for the problem
- Compare their performances with the optimal results

Mapping problem

- Given a KPN task graph
- Given an architecture graph and a deterministic routing algorithm
- Find the optimal mapping
- such that computation time and amount of communication is optimized

ILP formulation of the mapping problem

- A *task graph* $g_t = (V_t, E_t)$ is composed of tasks $t \in V_t$ and data dependencies $e \in E_t \subseteq V_t \times V_t$.
- An *architecture graph* $g_a = (V_n, E_n)$ is composed of processing nodes $n \in V_a$ and bidirectional communication links $l \in E_a \subseteq V_a \times V_a$.
- A *task binding* $\beta_t : V_t \rightarrow V_a$ is an assignment of tasks $t \in V_t$ to nodes $n \in V_a$.
- A *communication binding* $\beta_c : E_t \rightarrow E_a^i$ is an assignment of data dependencies $e \in E_t$ to paths of length i in the architecture graph g_a. A *path* p of length i is given by i-tuple $p = (l_1, l_2, ..., l_i)$.

ILP formulation of the mapping problem

- $path : (V_a, V_a) \rightarrow E_a^i$ is a function that implements a deterministic routing algorithm and returns a path between two given nodes. *Path set* P is the set of paths between all node pairs:

$$P = \{p_k : p_k = path(n_i, n_j), \forall n_i, n_j \in V_a \wedge n_i \neq n_j\}$$

- The task graph can be annotated with demand values where *demand* d_i on a data dependency $e_i \in E_t$, denotes the required bandwidth between the two tasks.
- The architecture graph can be annotated with capacity values where *capacity* on an architectural link $l_i \in E_a$, c_i, denotes the maximum bandwidth of the communication link between two architectural nodes.
- Core type set C consists of core types C_i and lists the types of cores available in a given NoC platform.

Minimization of the communication cost

Decision variables

$$X_{ij}^{NT} = \begin{cases} 1, & \text{if } t_j \in V_t \text{ is bound onto node } n_i \in V_a \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{ij}^{PE} = \begin{cases} 1, & \text{if } e_j \in E_t \text{ is mapped to } p_i \in P \\ 0, & \text{otherwise} \end{cases}$$

Minimization of the communication cost

Parameters

$$M_{ij}^{TE} = \begin{cases} 1, & \text{if } \exists t_k, e_j = (t_i, t_k) \in E_t \\ -1, & \text{if } \exists t_k, e_j = (t_k, t_i) \in E_t \\ 0, & \text{otherwise} \end{cases}$$

$$M_{ij}^{NP} = \begin{cases} 1, & \text{if } source(p_j) = n_i \\ -1, & \text{if } sink(p_j) = n_i \\ 0, & \text{otherwise} \end{cases}$$

$$M_{ij}^{PL} = \begin{cases} 1, & \text{if } l_j \in p_i \\ 0, & \text{otherwise} \end{cases}$$

Minimization of the communication cost

Constraints

Constraint 1 *(routing)*: Task mapping and communication binding are constrained with each other due to the routing algorithm implemented in the NoC.

$$X^{NT} M^{TE} = M^{NP} Y^{PE} \tag{1}$$

Constraint 2 *(task mapping)*: A task can be mapped exactly on one node.

$$X^{TN} \mathbf{1}_{|V_a|} = \mathbf{1}_{|V_t|} \tag{2}$$

Constraint 3 *(communication mapping)*: A data dependency can be mapped at most on one path.

$$Y^{EP} \mathbf{1}_{|P|} \leq \mathbf{1}_{|E_t|} \tag{3}$$

Minimization of the communication cost

Constraint 4 *(capacity)*: Total bandwidth demand on a link l_j should not exceed the capacity of the link c_j.

$$M^{LP} Y^{PE} d \leq c \tag{4}$$

Objective 1 *(communication cost)*: The total traffic on the links is the sum of all demands d_i on the links of the paths that arise according to a given mapping:

$$\min: \; d^T \; Y^{EP} M^{PL} \, \mathbf{1}_{|E_a|} \tag{5}$$

Minimization of the total computation time

Parameters

$$M^{TC}_{cap\,ij} = \begin{cases} 1, & \text{if } C_j \in C \text{ is capable of realizing task } t_i \in V_t \\ 0, & \text{otherwise} \end{cases}$$

$$T^{TC}_{cap\,ij} = \begin{cases} \text{completion time of } t_i \text{ on } C_j, & \text{if } M^{TC}_{cap\,ij} = 1 \\ 0, & \text{if } M^{TC}_{cap\,ij} = 0 \end{cases}$$

$$M^{NC}_{ij} = \begin{cases} 1, & \text{if } n_i \in V_a \text{ is of core type } C_j \in C \\ 0, & \text{otherwise} \end{cases}$$

Minimization of the total computation time

Constraints

Constraint 5 *(capability)*: All tasks should be mapped on cores that are capable of implementing those tasks.

$$M^{TC} = X^{TN} M^{NC} \leq M^{TC}_{cap} \tag{6}$$

Minimization of the total computation time

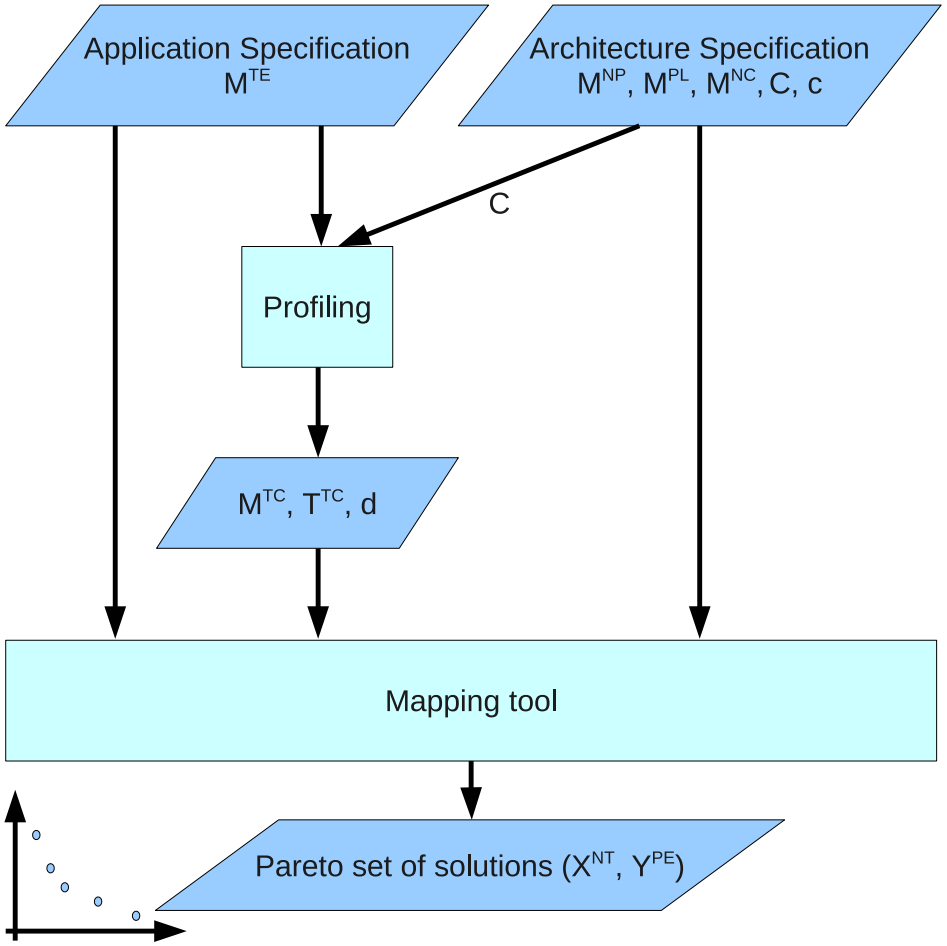
Objective 2 *(total execution time)*: We calculate the total computation time of the application by finding the maximum of the sum of the execution times of tasks mapped on the same core.

$$\min: \; max(T^N) = max(X^{NT}(((X^{TN}M^{NC}) \; . \; T^{TC}_{cap}) \; \mathbf{1}_{|C|})) \qquad (7)$$

- We apply some linearization techniques to this formula for $max()$ and $x_{ij} * x_{kl}$
- The analytical model for the adopted objectives is valid
  - for acyclic KPN applications
  - in cases where communication is faster with respect to computation
  - when the network is not overloaded (no congestion)

The mapping tool

- based on *IBM® ILOG® CPLEX® API*
- multi-objective ILP problem solved with ε-constraint method



Optimal Task Remapping

New constraints

Constraint *(faulty core)*: Given a faulty node n_f, a new constraint is added to the ILP formulation that forbids mapping of tasks on the faulty node n_f.

$$\sum_{j=1}^{|V_t|} X_{fj}^{NT} = 0 \tag{8}$$

Constraint *(migrate only tasks on the faulty core)*: Given a faulty node n_f and an initial task mapping M^{NT}, a new constraint is added to limit the reconfiguration just to the tasks that are running on the faulty node n_f.

$$X_{ij}^{NT} = M_{ij}^{NT},\ 1 \leq i \leq |V_a|,\ 1 \leq j \leq |V_t|,\ i \neq f \tag{9}$$

Optimal Task Remapping

- Why not encode optimal remapping solutions for every fault scenario?
- number of scenarios, N

$$N = \sum_{i=1}^{|V_a|} \binom{|V_a|}{i} - 1 = 2^{|V_a|} - 1$$

- number of bits, B

$$B = (2^{|V_a|} - 1)\, p\, |V_t|\, \lceil log(|V_a|) \rceil$$

- For $|V_a| = 9$, $|V_t| = 12$, $p = 5$, we have $B = 14.97$ Kbytes
- We may need heuristics!

Center of Gravity method (CoG)

Idea

We transform the problem to finding the center of gravity of masses by considering the weights of communication as the masses of the peer tasks.

$$coord_i = \frac{\sum_{t_j \in peers(t_i)} coord(\, map(t_j)\,)\; weight(t_j, t_i)}{\sum_{t_j \in peers(t_i)} weight(t_j, t_i)},\; t_i \in L_f$$

$$coord(n_i) = \lfloor coord_i + (0.5, 0.5) \rfloor$$

- considers only communication

Nonidentical Multiprocessor Scheduling Heuristics (NMS)

Idea

The objective regarding computation is equivalent to the scheduling of independent tasks on nonidentical processors in order to minimize the makespan.

- Three heuristics: NMS-A, NMS-B, NMS-C
- Different complexities: $O(n)$, $O(n \log n)$, $O(n^2)$
- No absolute winner
- considers only computation

NMS Heuristics

NMS-A

The task $t_i \in L_f$ is scheduled on the core that minimizes its finishing time.

NMS-B

For each task $t_i \in L_f$, Algorithm NMS-B first orders the tasks in L_f according to decreasing $min\{T^{TN}_{cap\ ij} : 1 \leq j \leq |V_a|\}$, and then calls Algorithm NMS-A.

NMS-C

This algorithm iteratively schedules the tasks by choosing a task from L_f that gives the least finishing time.

Localized NMS Heuristics (LNMS)

Idea

We limit the region of nodes where we employ the NMS heuristics.

- considers both communication and computation
- apply NMS heuristics on a region: LNMS-A, LNMS-B, LNMS-C
- region is parametrized in size and centered around center of gravity

Case study

- 15 seconds long video with resolution 704×576 pixels and 25 fps

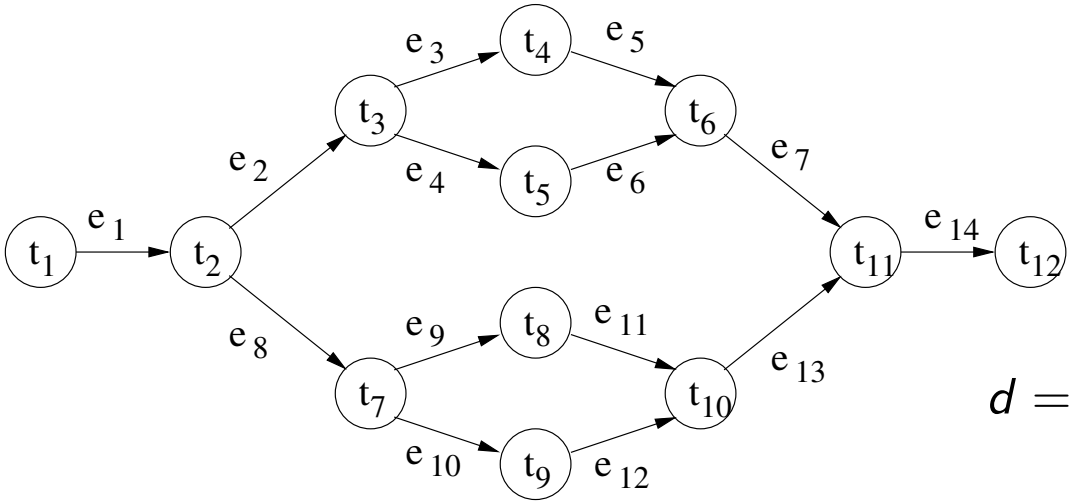


Figure: MPEG-2 Decoder KPN graph

$$d = \begin{pmatrix} 1.0 \\ 34.6 \\ 28.1 \\ 28.1 \\ 28.1 \\ 28.1 \\ 65.0 \\ 34.6 \\ 28.1 \\ 28.1 \\ 28.1 \\ 28.1 \\ 65.0 \\ 15.2 \end{pmatrix} \text{(in MBps)}$$

Case study

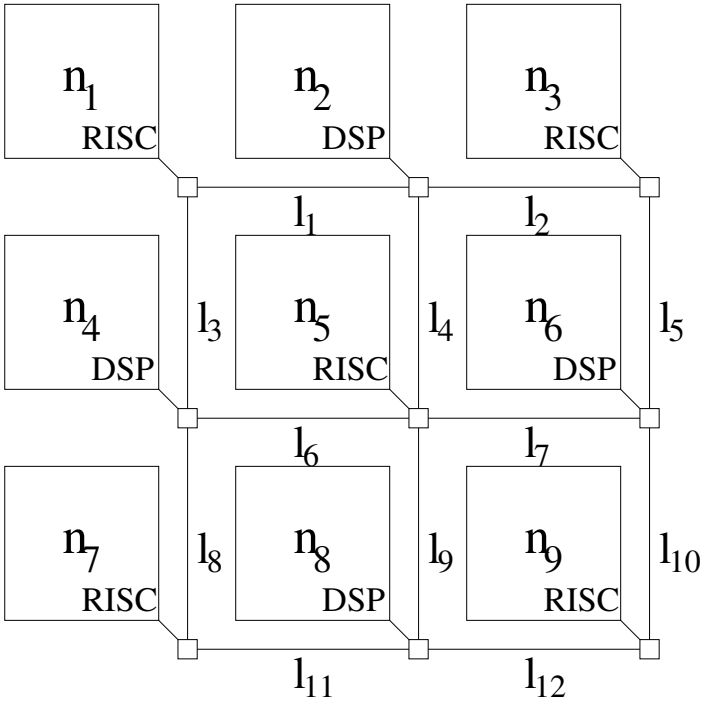


Figure: The given 3x3 NoC architecture

$$c_i = 100 \text{ MBps}, \; 1 \leq i \leq |E_a|$$

$$C = \{C_1, C_2\} = \{\text{RISC}, \text{DSP}\}$$

$$M_{cap}^{TC} = \mathbf{1}_{12\times 2}$$

$$T_{cap}^{TC} = \begin{pmatrix} 0.13 & 0.20 \\ 6.68 & 8.52 \\ 0.06 & 0.04 \\ 2.00 & 1.25 \\ 2.00 & 1.25 \\ 0.05 & 0.04 \\ 0.06 & 0.04 \\ 2.00 & 1.25 \\ 2.00 & 1.25 \\ 0.05 & 0.04 \\ 12.33 & 8.51 \\ 0.18 & 0.30 \end{pmatrix} \text{(in seconds)}$$

Results

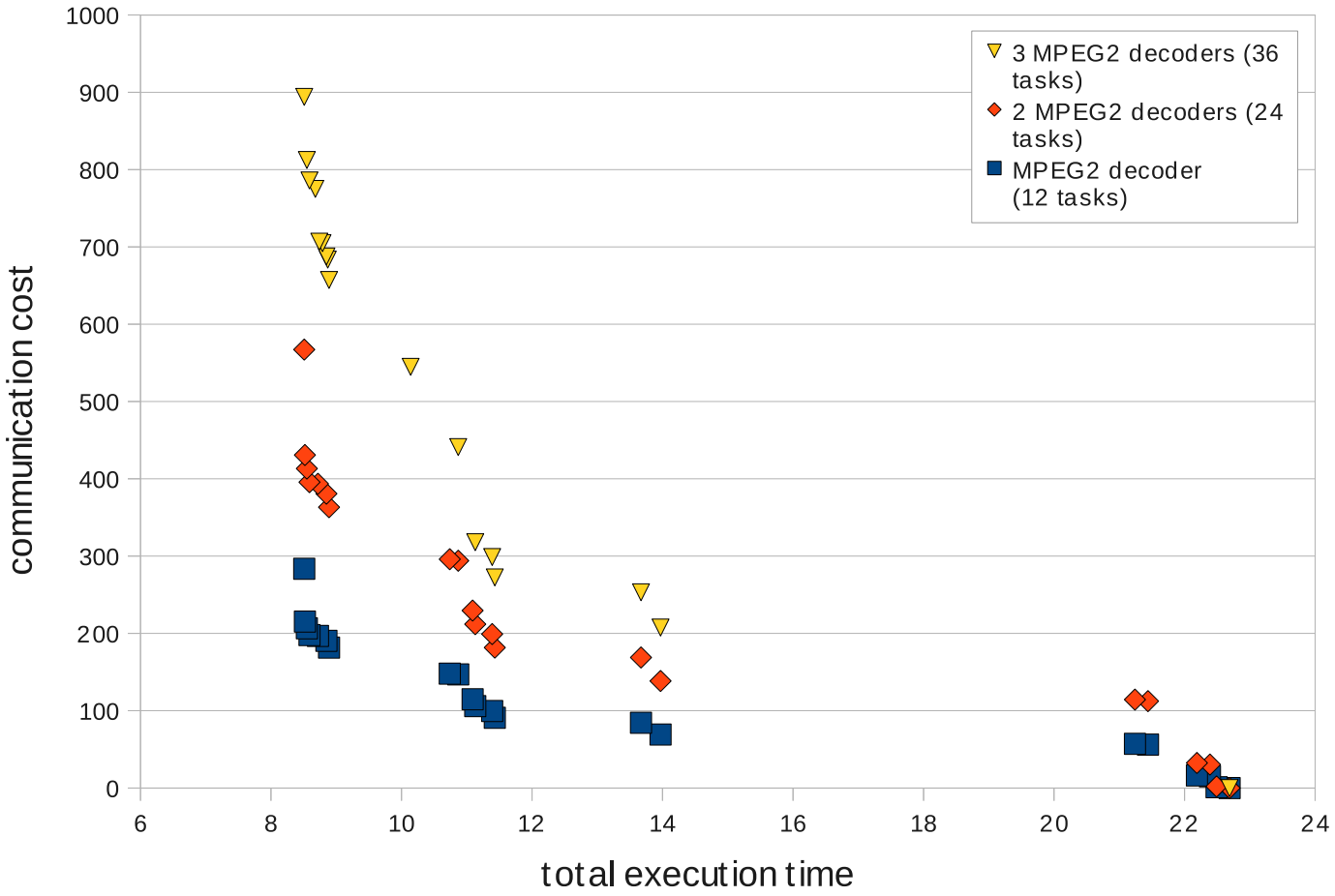


Figure: Pareto curves for optimal mapping of three applications with 12, 24 and 36 tasks

Results (MPEG2 decoder - 3x3 NoC - n_5 faulty)

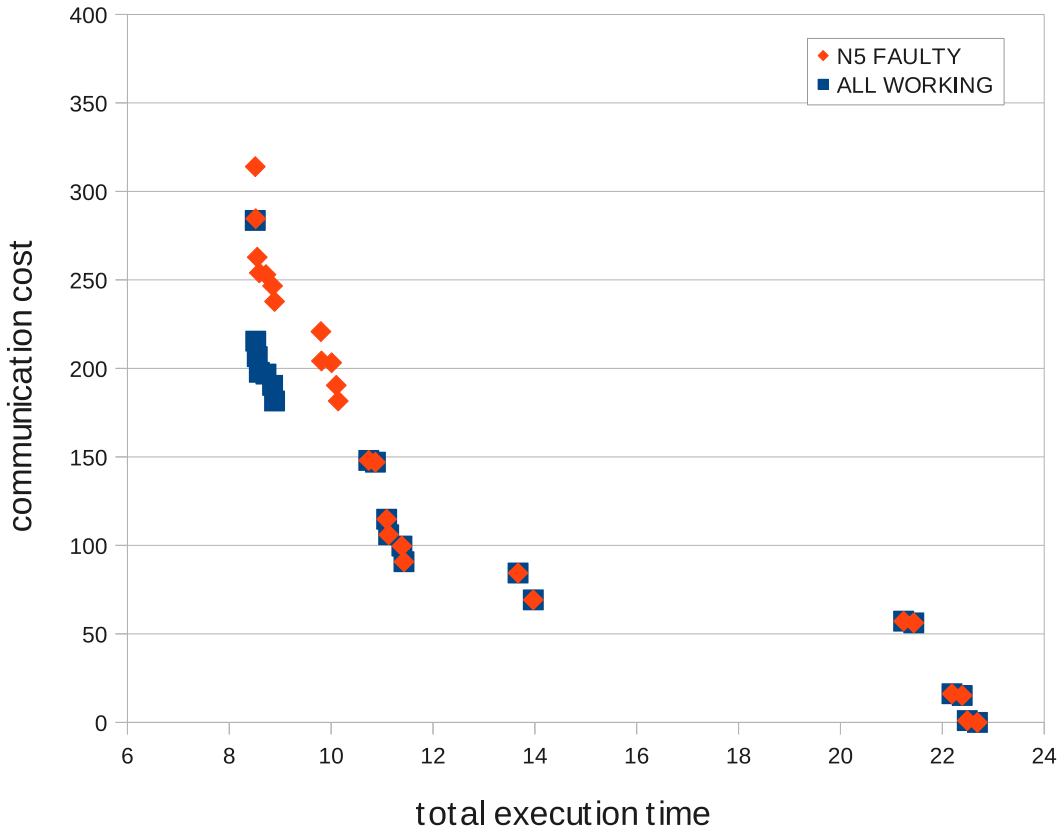


Figure: Pareto curves for optimal mappings and optimal remappings (unlimited task migration)

Results (MPEG2 decoder - 3x3 NoC - n_5 faulty)

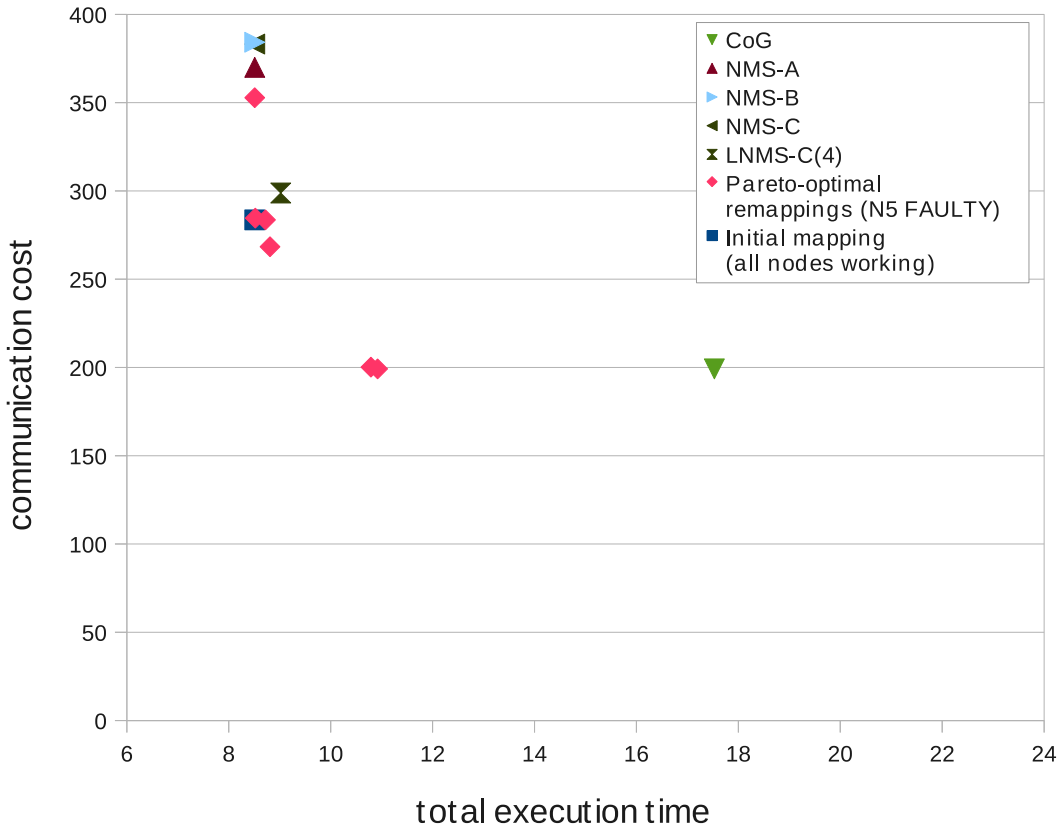


Figure: Comparison of results between heuristics, optimal remappings and the initial mapping (limited task migration)

Results (MPEG2 decoder - 3x3 NoC - all single faults)

faulty node	mapping	exe.time (obj. 1)	com.cost (obj. 2)	degradation	
				obj. 1	obj. 2
none	initial	8.51	283.6	-	-
n_1	Pareto1	8.51	305.2	1.00	1.08
	Pareto2	8.55	296.4	1.00	1.05
	Pareto3	9.05	261.8	1.06	0.92
	Pareto4	9.80	205.6	1.15	0.73
	Pareto5	11.09	184.0	1.30	0.65
n_2	Pareto1	8.51	413.6	1.00	1.46
	Pareto2	16.44	298.8	1.93	1.05
n_5	Pareto1	8.51	352.8	1.00	1.24
	Pareto2	8.52	284.6	1.00	1.00
	Pareto3	8.72	283.6	1.02	1.00
	Pareto4	8.81	268.4	1.04	0.95
	Pareto5	10.79	200.2	1.27	0.71
	Pareto6	10.92	199.2	1.28	0.70

Figure: Degradation achieved by Pareto-optimal limited remappings for all single fault scenarios

faulty node	Approach	exe.time (obj. 1)	com.cost (obj. 2)	degradation	
				obj. 1	obj. 2
none	initial	8.51	283.6	-	-
n_1	CoG	10.99	296.4	1.29	1.05
	NMS-A	8.51	603.8	1.00	2.13
	NMS-B	8.51	603.8	1.00	2.13
	NMS-C	8.51	482.6	1.00	1.70
	LNMS-C(4)	8.51	314	1.00	1.11
n_2	CoG	16.44	298.8	1.93	1.05
	NMS-A	8.51	413.6	1.00	1.46
	NMS-B	8.51	413.6	1.00	1.46
	NMS-C	8.51	413.6	1.00	1.46
	LNMS-C(4)	8.51	413.6	1.00	1.46
n_5	CoG	17.53	199.2	2.06	0.70
	NMS-A	8.51	370.0	1.00	1.30
	NMS-B	8.51	384.2	1.00	1.35
	NMS-C	8.51	383.2	1.00	1.35
	LNMS-C(4)	9.02	298.8	1.06	1.05

Figure: Degradation achieved by proposed heuristics for all single fault scenarios

Conclusions

- We proposed an ILP formulation to the task mapping problem onto heterogeneous NoC multiprocessors with deterministic routing optimizing the total execution time and total network traffic.
- We proposed ILP-based optimal solution to the remapping problem due to faulty nodes.
- We proposed heuristics and evaluated them in comparison to the optimal remapping results.
- ILP is not scalable but it makes sense to use it for the remapping problem.

Future work

- get results for bigger task and node sizes
- implement on the MADNESS NoC platform

Università della Svizzera italiana

Advanced Learning and Research Institute ALaRI

Madness

http://www.madnessproject.org

Online Task Remapping Strategies for Fault-tolerant Network-on-Chip Multiprocessors

Onur Derin, Deniz Kabakcı, Leandro Fiorin

ALaRI
Faculty of Informatics
University of Lugano
Lugano, Switzerland

derino@alari.ch

NOCS'11 - Pittsburgh
May 3, 2011