Real-Time Low-Power Task Mapping in Networks-on-Chip

M. Norazizi Sham Mohd Sayuti, Leandro Soares Indrusiak
Real-time Systems Group, Department of Computer Science,
University of York
Email: {azizi, lsi}@cs.york.ac.uk

Abstract—**Many state-of-the-art approaches to power minimisation in Networks-on-Chip (NoC) are based on the reduction of the communication paths taken by packets over the interconnect. This is often done by optimising the packet routing, the allocation of tasks that produce and consume those packets, or both. In all cases, the optimisation affects the timeliness of the packets, because changes will occur in the way resources are shared at the platform cores (as tasks are reallocated) and NoC links (as packet routes are changed). In this paper, we propose an optimisation technique that is able to minimise power dissipation without sacrificing timing constraints, thus suitable to systems with hard real-time requirements. It is based on a Genetic Algorithm (GA) that evolves chromosomes representing the mapping of tasks to cores, guided by a multi-objective fitness function that combines power estimation macromodels and real-time schedulability analysis.**

I. Introduction

This paper addresses the design of embedded multicore systems based on Networks-on-Chip (NoC), aiming to find system configurations that minimise power dissipation but still meet real-time performance requirements at all times. While generally scalable and flexible, NoCs are complex to analyse when it comes to evaluate performance guarantees and power dissipation. This becomes critical when addressing embedded systems with hard real-time constraints, which is our focus in this paper. In such systems, data packets sent over the NoC have strict deadlines, i.e. the latest point in time when they can be delivered at their respective destination cores. A deadline miss is considered a failure, just as bad as a packet being dropped or corrupted. The strictness of the performance constraints limits significantly the application of conventional low power techniques in NoCs. State-of-the-art approaches in energy-aware task mapping affect the timing behaviour of the system for the sake of minimising energy dissipation [1] [2]. For example, a common technique is to map communicating tasks to neighbouring cores, thus reducing the number of network hops a packet must go through. While minimising energy dissipation, such optimisation also changes the way resources are shared in the system, which consequently affects the response time of the tasks running on cores and packets over the NoC.

In the following sections, we present an optimisation approach based on a genetic algorithm (GA) which was specifically customised to statically map tasks onto NoC-based multicore systems. It uses real-time schedulability analysis to constrain the way tasks are mapped, so that the timing requirements of the system (i.e. deadlines of tasks and packet flows) can be met at all times. We also integrate a power estimation macromodel into the GA's fitness function, so that it can be guided towards more energy-efficient solutions as it explores alternative task mappings.

II. Background

A. Application Model

This paper aims to optimise the mapping of application tasks onto cores of a NoC-based platform. Following a common practice in real-time systems design, we use an application model to obtain a design-time characterisation of the computation and communication load imposed by each application. An application model consists of a set of tasks, each of them is characterised as a tuple $Task = \{C, T, V\}$, where C is its maximum execution time, T is its period (i.e. the minimum amount of time between successive invocations of that task), and V is its priority. Tasks communicate with each other by sending messages, each of them is described as a tuple $Msg = \{Src, Dst, V, F\}$, where Src is the task that sends it, Dst is its destination task, V is its priority, and F its length.

We assume that a message is sent at the end of execution of its source task, so the period of a message is equal to the period of its source. We also assume that the deadline D of a task or a message is equal to its period. A system is then deemed schedulable if all its tasks and messages have a worst case response time that is less or equal to their respective periods. That way, developers can define periods to accommodate application-specific timing constraints, and by ensuring schedulability they can guarantee that all tasks and messages will always successfully complete before their succeeding release.

B. Network-on-Chip Platform

This paper considers homogeneous platforms consisting of a number of processing cores interconnected with a wormhole-switching NoC. Each core can execute multiple tasks, which are scheduled using priority preemptive arbitration. Tasks communicate with each other by sending and receiving messages, and such messages are packetised and sent over the NoC if the source and destination tasks are not mapped to the same core. Every periodic transfer of a packet over the NoC is referred in this paper as a packet flow. A flow will have application-specific characteristics (e.g. period of its source task, length of message) as well as platform-specific ones (e.g. route over the NoC, number of flits needed to carry the respective message plus control information).

The NoC architecture considered here has routers that use deterministic routing and priority preemptive virtual channels [3]. This means that packets of a given flow will always follow the same route (which is known at design time), and that packets with lower priority are preempted when they compete against higher priority packets for the same NoC link.

C. Power Estimation Macromodel

The macromodel proposed by [4] can be used to estimate the total power dissipated by a NoC for transferring a given packet. According to that macromodel, each network component (i.e. network interface - NI, router and link) dissipates power to transmit a packet along the specified route. Equation (1) calculates the dissipation by each component, and differs from the one appearing in [4] only because we assume that the power dissipated by a router to transmit a header flit or a payload flit is the same:

$$P_m = 2(f+1)P_{ni} + (h+1)(f+1)P_r + h(f+1)P_l \qquad (1)$$

where P_{ni} is the power dissipated by a NI to transmit a flit, P_r is the power dissipated by the router to transmit a flit, P_l is the power dissipated by the link to transmit a flit, f is the number of flits in the packet, and h is the number of hops of its route.

D. Real-Time Schedulability Analysis

We assume that each core has its own priority-ordered queue to schedule tasks, so classical schedulability analysis for single processor [5] can be applied to determine the worst-case response time r_i of each task i, as shown in equation (2). The function hp denotes the set of all higher priority tasks running on the same core as task i.

$$r_i^{n+1} = C_i + \sum_{\forall_{Task_j} \in hp(i)} \left\lceil \frac{r_i^n}{T_j} \right\rceil C_j \qquad (2)$$

In [3], Shi et al propose a schedulability analysis that enables the calculation of the worst-case latency of each packet flow in a wormhole NoC. That analysis finds the upper bound of the interference suffered by a given flow, caused by a set of flows S_i^D that compete for the same NoC links and that can preempt it. As shown in equation (3), the worst-case latency R of a flow is the sum of its basic latency L_i (which is the latency of the flow when there is no contention) plus all the interference caused by each of the preempting flows (which is proportional to their basic latency L_j).

$$R_i^{n+1} = L_i + \sum_{\forall_{Flow_j} \in S_i^D} \left\lceil \frac{R_i^n + J_j^R + J_j^I}{T_j} \right\rceil L_j \qquad (3)$$

Equation (3) also takes into account that the release jitter J_j^R and the interference jitter J_j^I will affect the worst-case latency of a flow. Since we assume that a flow is released immediately after the execution of its source task, J_j^R is equal to the response time of its source task r_j. Finally, from [3] we have that $J_j^I = R_j - L_j$. Based on the $D \leq T$ assumption, an application is deemed schedulable if for each task i, $r_i \leq D_i$ and for each flow i, $r_i + R_i \leq D_i$.

E. Mapping Optimisation with Genetic Algorithms

Genetic algorithms have been used to optimise the mapping of application tasks to multicore platforms. Such algorithms use one or more fitness functions to guide a random search towards solutions of increasing fitness. All approaches in the literature use fitness functions based on simulation or analytical methods.

Different mappings produce different performance and energy dissipation metrics. By mapping communicating tasks closer to each other, it is possible to decrease the power dissipated by the NoC but the response time of the tasks and messages can be affected because of the increased contention over the shared links. Hence, considering multiple fitness functions is important so such trade-offs can be addressed.

In [1], a GA was used to solve the mapping problem, aiming to minimise the amount of communication delay and the average power consumption. It used the SPEA2 GA [6] to explore the mapping space, and used a NoC simulator to determine the fitness of every solution. A severe limitation of that approach was the time required to evaluate each solution with the simulator, specially because one needs a long simulation time to have good confidence on the figures of average delay and power consumption (i.e. long runs using a high-accuracy simulator). Moreover, worst-case scenarios can easily be missed using this technique, as it is not trivial to find the right stimulus for such scenarios to arise. Such limitations prevent the use of the technique for early design space exploration (as it requires fast evaluation), and makes it unsuitable for applications with real-time constraints (as there is no guarantee that worst-case scenarios will ever be simulated).

Other works [7] [8] [9] applied analytical methods as fitness functions, which enable much faster exploration of the mapping space, but none of them addressed the problem of mapping tasks and messages with hard real-time constraints.

The only approaches that are able to find mappings for priority-based hard real-time systems are based on single-objective genetic algorithms [10] [11]. While they are able to find fully schedulable mappings, they are oblivious to the changes in power dissipation by different mapping alternatives.

III. Proposed Mapping Optimisation

In the subsections below, we propose an optimisation pipeline based on a multi-objective genetic algorithm (MOGA) that integrates two fitness functions, aiming to find task mappings that meet hard real-time application constraints while minimising power dissipation over the NoC.

A. Genetic Algorithm

We chose NSGA-II [12] as the underlying GA that will be the basis of our optimisation pipeline. This is mainly due to its non-dominated sorting mechanism, which makes it amenable to problems with multiple objectives such as ours; besides of its configurability and efficient implementation. This key point will become clearer once we describe the complete optimisation pipeline proposed here.

In general, a GA works by manipulating chromosomes which represent an individual solution to the problem we are trying to optimise. In our case, a chromosome must represent a specific task mapping, and Fig. 1 shows one possible way to encode that information: the indexes represent n tasks, and under each index the chromosome stores the number of the processing core onto which the task is mapped (e.g T_1 is mapped to core 1, T_3 to core 9). Each part of the chromosome is called a gene, e.g. a task index and its respective core number in Fig. 1.

Our GA optimisation pipeline (Fig. 2) starts with an initial parent population, represented by their chromosomes, which can be randomly created (i.e. randomly select the value of each gene of each chromosome). It then creates an offspring population by operating over the parent population. Finally, it applies one or more fitness functions which will rank all chromosomes of the combined population and guide the selection of the chromosomes which will be allowed into the next generation.

It is at the final stage of the pipeline that NSGA-II's non-dominated sorting plays an important role. When sorting over multiple criteria, as in our case the real-time schedulability and total energy dissipation of a particular mapping, it is common to find solutions that cannot be easily compared. For instance, a mapping that is fully schedulable, but dissipate more energy than another mapping that is only partially schedulable.

The proposed optimisation relies on the notion of Pareto-dominance [13] supported by NSGA-II, which states that a multi-criteria solution dominates another if it is better in at least one criterion and no worse in all the others. This allows us to partition the combined population in non-dominated sets. The chromosomes in the first non-dominated set are not dominated by any chromosome within the parent or offspring generations. Once the set is found, it is labelled with a non-domination level 1 and all the chromosomes in this set inherit this level as their fitness value. Then, this non-dominated set is excluded from the process of determining the next non-dominated set. This process is repeated for the rest of the individuals in the combined population until all of them are ranked with their respective non-domination levels as shown in Fig. 3. After this process, a technique called elitism is applied to transfer only the best chromosomes of a given generation to the subsequent generation.

As mentioned above, GAs create an offspring population by applying specific operations over the parent population. Our optimisation pipeline considers the three main types of operators found in GAs: selection, crossover and mutation.

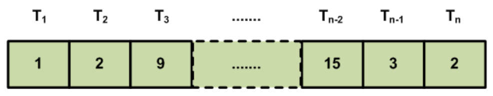


Fig. 1. Chromosome structure

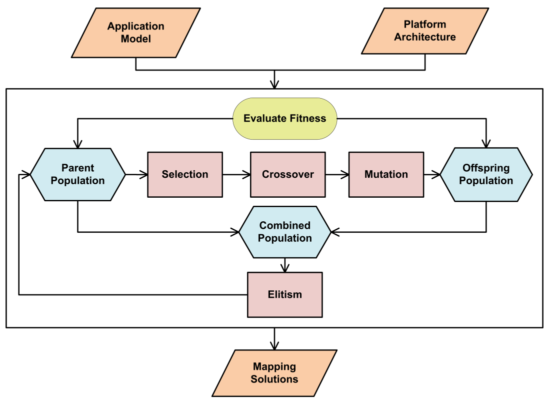


Fig. 2. Optimisation pipeline

The selection operator is responsible for selecting which parent chromosomes are used to produce the offspring generation. Since genes of offspring are inherited from their parents, the selection operator must favour the best individuals in the population. We use binary tournament selection ($k = 2$), as shown in Fig. 4 two possible parents are randomly chosen and then compared in terms of fitness, i.e their non-domination level [12]. Only winner of the tournament (i.e. the one with the lowest non-domination level) is selected for the mating process. The procedure has to be repeated to select the other mating parent.

The mating process is a single-point crossover performed on the parents by exchanging a part of their chromosome to produce two new offspring. The amount of genes that each parent contributes is determined by a crossover point which is selected randomly (Fig. 5).

We also use a mutation operator to introduce additional variations to the chromosomes of the offspring population. The mutation operator works by randomly changing individual genes of the offspring according to a given probability.

B. Objective Functions

The focus of our optimisation pipeline is to find task mappings that present a good trade-off between two objectives: meeting timing constraints and minimising power dissipation. The first objective (Obj_1) is to minimise the total number of unschedulable tasks (Ut) and flows (Uf) as shown in Eq. (4). The worst-case response time r_i of each task is calculated based on equation (2), while the worst-case latency R_i of each flow is calculated using equation (3). Tasks are unschedulable if their worst-case execution time exceed their deadline, and flows are unschedulable if their worst-case latency exceeds the deadline of their source task (which is the end-to-end deadline all tasks and flows must meet). Therefore, it is possible to have

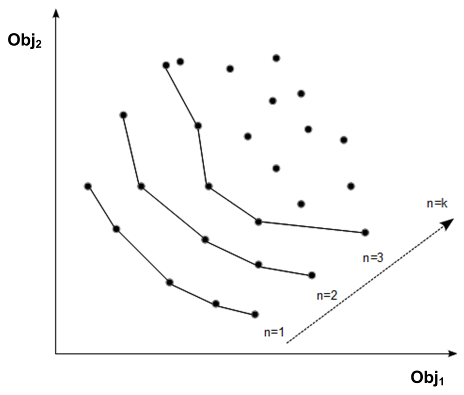


Fig. 3. Non-domination levels

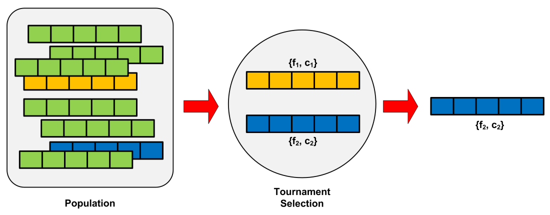


Fig. 4. Binary tournament selection

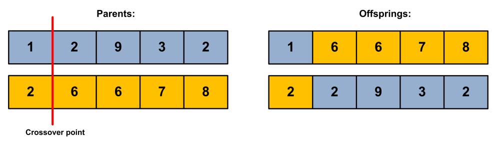


Fig. 5. Single-point crossover

mappings where a task meets its deadline but its respective flows do not.

The second objective (Obj_2) considered here is the minimisation of the total power dissipated when transmitting packets over the NoC, as shown in equation (5). This metric is calculated using equation (1). Our optimisation pipeline does not take into account the energy dissipated by the execution of tasks by each individual cores, as that metric does not contribute to the ranking of alternative mappings in terms of the overall energy dissipation of the system (i.e. communication, all cores will dissipate roughly the same amount of energy to execute a particular task). This situation would be of course different if our optimisation would also include thermal balance as one of its objectives, but this is left as future work.

$$Obj_1 = \min(\sum_{i=1}^{k} Ut_i + \sum_{i=1}^{l} Uf_i) \tag{4}$$
$$s.t.$$
$$\forall Task_i, r_i > D_i \Rightarrow Ut_i = 1$$
$$\forall Flow_i, r_i + R_i > D_i \Rightarrow Uf_i = 1$$

$$Obj_2 = \min(\sum_{m=1}^{l} P_m) \tag{5}$$

IV. Experimental Work

A. Case Studies

Two benchmarks were used as case studies in this paper. The first benchmark is based on a realistic autonomous vehicle application (AV) which consists of 33 tasks and 38 messages, while the second benchmark is a synthetic application (SA) consisting of 50 real-time tasks and 50 real-time messages. The synthetic application was created to have shorter inter-arrival intervals (periods) of tasks and messages and shorter slacks, so finding schedulable mappings is harder. Optimisation was performed to find feasible mapping solutions in 4x4 and 5x5 2D-mesh topology NoC platforms with 16-bit links, XY routing policy and 2-position buffers per virtual channel. Each packet flow has the same priority value as the task which releases it.

B. Evaluation Results

We use the benchmarks above to compare the proposed multi-objective GA (MOGA) with a number of baselines: a naive random mapper, a nearest neighbour (NN) mapper and the single-objective GA (SOGA) proposed in [11].

Our experimental hypothesis is that the mapping solutions found by our MOGA will be as good as the solutions produced by the SOGA (or better) in terms of meeting timing constraints, and always better in terms of power dissipation. We expected that the second part of the hypothesis would be easy to show, because the single-objective GA does not optimise power dissipation. The challenging part is to show that the MOGA is able to quickly converge towards fully schedulable solutions, which is not straightforward as it also has to keep many low power mappings within the population at every generation. This has been achieved by careful parametric analysis, which included the ranges shown in Table I.

By providing evidence that the above experimental hypothesis is valid, we also aim to show that both GA-based mapping optimisation can produce mappings that are far superior than the naive and NN mappers.

Fig. 7 plots the improvement of the fitness of the mappings found by each of the GA-based optimisation pipelines (MOGA and SOGA). The horizontal axis represents the evolution across generations, while two vertical axes allow us to plot the number of unschedulable tasks and flows (left axis, labelled as UTF) and the total energy dissipation across the NoC (right axis, labelled as POW, normalised by the power dissipated by a single flit over a single hop) for the best mapping of each generation. In single-objective optimisation, solutions can be totally ordered, so the best solution is evident. However, in multi-objective optimisation there may be no single best solution, as the set of non-dominated solutions may have a cardinality larger than one, and all members of that set can be considered as the best solution. Therefore we plotted Fig. 7 with the solution having the lowest number of unschedulable tasks and flows. If more than one solution has the same value, the secondary metric is then used and the solution with the lowest power dissipation is selected.

TABLE I
SELECTED GA PARAMETER VALUES

GA Parameters	Values
Population Size	100, 1000
Crossover Rate	0.5, 0.8
Mutation Rate	0.01, 0.001
Max Generations	50, 500

Fig. 7 shows that the best overall solution found by the SOGA is able to meet the real-time constraints of the AV benchmark on both 4x4 and 5x5 NoCs and the SA on the 5x5 NoC only. The same result is also achieved by the MOGA. Except for the AV benchmark over the 5x5 platform, SOGA converges faster than MOGA. However, the SOGA's total power dissipation did not decrease consistently across generations (red and the blue lines). Although there are several times when some reduction occurs, the trend is not consistent in the long run. On the other hand, the proposed MOGA is able to obtain mappings that meet all real-time constraints and improves consistently the total power dissipation (green and purple lines). This validates our experimental hypothesis for the chosen benchmarks, as the MOGA was able to produces solutions that are able to meet the real-time constraints for all the scenarios that the SOGA has succeeded, but always with lower power dissipation.

The plot in Fig. 8 shows how the level 1 non-dominated set converges across generations for each algorithm, using a population of 100 chromosomes. Each line represents the non-dominated set of the population at different generations (1st, 100th and 500th generations). The green lines represent the non-dominated sets optimised with MOGA, showing better convergence towards the optimal region of the solution space (lower-left corner, where fully schedulable low power solutions are found). Both GAs are able to find a schedulable mapping (i.e. touching the vertical axis) at 100 and 500 generations, but again the mappings found by the MOGA have much lower power dissipation. As shown in the plot, there is a significant difference in power dissipation, which highlights how much quality improvement can be made when both objectives are taken into account. To show the influence of the parameters described in Table I, we compared different configurations of the proposed MOGA and plotted the results in Fig. 9. Each line represents the non-dominated set produced by each MOGA configuration at the first and last generations. For both benchmarks, the performance of the MOGAs configured with the parameter set represented by the green and purple lines is superior than the other options we considered (brown and cyan lines). Interestingly, the two best configurations have the same mutation rate but only differ in their crossover rate. We then compare the overall percentage of schedulable tasks and flows and the overall power dissipation by the best mapping found with that MOGA configuration with the best mappings found by each of the baselines, once more validating our experimental hypothesis for both benchmarks (Fig. 6).

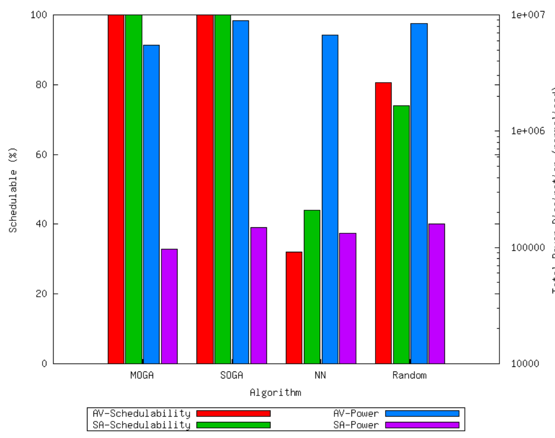


Fig. 6. Overall comparison against baseline

V. Conclusion

This work has proposed a novel optimisation approach for Networks-on-Chip which is able to explore a wide space of task allocation alternatives and can converge towards task mappings that meet hard real-time constraints and have minimal power dissipation. The main contributions of the paper include: (i) the integration of analytical methods for real-time analysis and power estimation as fitness functions of a MOGA; and (ii) extensive experimental work to support the parametric analysis of the proposed MOGA-based optimisation pipeline, and to show that it fully dominates the best known single-objective genetic mapping algorithm for hard real-time NoC systems.

References

[1] G. Ascia, V. Catania, and M. Palesi, "Multi-objective mapping for mesh-based noc architectures," in *CODES+ISSS*, 2004, pp. 182 – 187.

[2] J. Hu and R. Marculescu, "Energy-aware mapping for tile-based noc architectures under performance constraints," in *Design Automation Conference*, 2003, pp. 233 – 239.

[3] Z. Shi, A. Burns, and L. Indrusiak, "Schedulability analysis for real time on-chip communication with wormhole switching," *IJERTCS*, vol. 1, no. 2, pp. 1–22, 2010.

[4] M. Palesi, G. Ascia, F. Fazzino, and V. Catania, "Data encoding schemes in networks on chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 5, pp. 774 –786, 2011.

[5] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," *Software Engineering Journal*, vol. 8, no. 5, pp. 284 –292, 1993.

[6] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN 2001*, 2002, pp. 95–100.

[7] W. Zhou, Y. Zhang, and Z. Mao, "Pareto based multi-objective mapping ip cores onto noc architectures," in *APCCAS*, 2006, pp. 331 –334.

[8] R. K. Jena and G. K. Sharma, "A multi-objective evolutionary algorithm based optimization model for network-on-chip synthesis," in *International Conference on Information Technology*, 2007, pp. 977 –982.

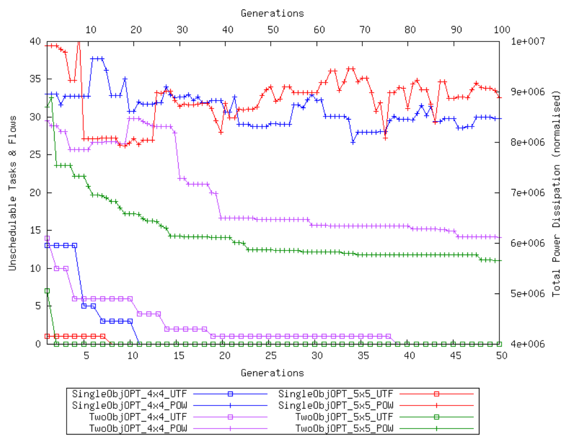
[9] N. Nedjah, M. Silva, and L. Mourelle, "Customized computer-aided application mapping on noc infrastructure using multi-objective optimization," *JSA*, vol. 57, no. 1, pp. 79 – 94, 2011.

[10] P. Mesidis and L. Indrusiak, "Genetic mapping of hard real-time applications onto noc-based mpsocs: A first approach," in *ReCoSoC*, 2011, pp. 1 –6.

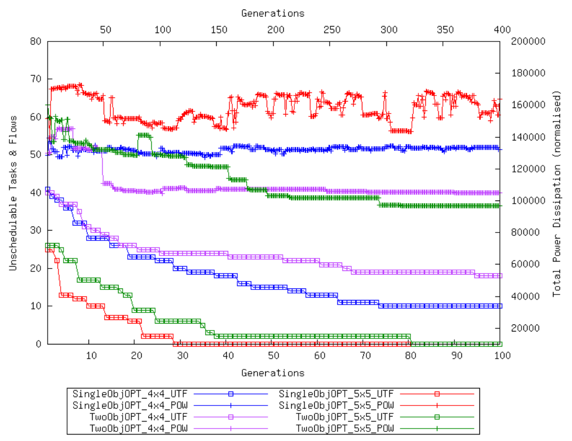
[11] A. Racu and L. Indrusiak, "Using genetic algorithms to map hard real-time on noc-based systems," in *ReCoSoC*, 2012, pp. 1–8.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[13] V. Pareto, *Cours d'economie politique*. Librairie Droz, 1964.

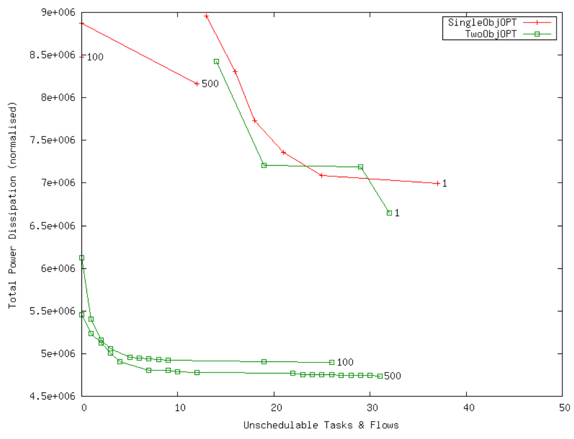


(a) Autonomous vehicle application

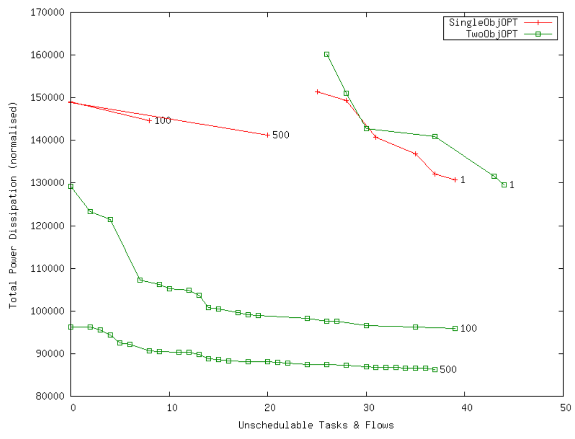


(b) Synthetic application

Fig. 7. The best solution convergence over generations with crossover rate 0.5 and mutation rate 0.01

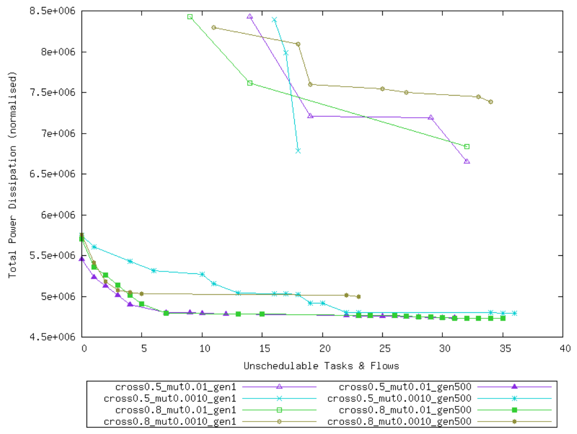


(a) Autonomous vehicle application, 4x4 platform

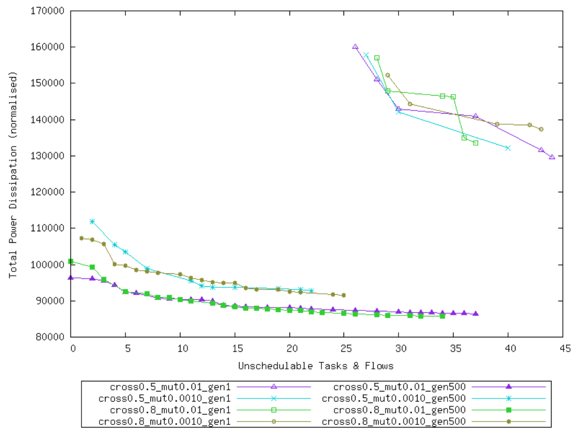


(b) Synthetic application, 5x5 platform

Fig. 8. The non-dominated set convergence at selected generations (1st, 100th and 500th) with crossover rate 0.5 and mutation rate 0.01



(a) Autonomous vehicle application, 4x4 platform



(b) Synthetic application, 5x5 platform

Fig. 9. The non-dominated set produced with different GA configurations