Mapping Real Time Applications on NoC Architecture with Hybrid Multi-objective PSO Algorithm

A.Benyamina
Invited Searcher at INRIA/FUTUR France
Laboratoire d'Informatique d'ORAN-LIO
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie
Email: benyanabou@yahoo.fr

B.Beldjilali
Laboratoire d'Informatique d'ORAN-LIO
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie
Email: bouzianebeldjilali@yahoo.fr

S.Eltar and K.Dellal
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie

***Abstract*—Many tools will be required to develop a NoC architecture for a specific application. A tool which can map and schedule an application or a set of applications to a given NoC architecture will be essential and must be able to satisfy many relative trade-offs (real-time, performance, low power consumption, time to market, re-usability, cost, area, etc). This paper targets a very important sentence in the cycle development embedded systems, the design software. Our problem is related to the issue of solving the problem of placing application in a network on chip under the constraints of load balancing, bandwidth, available memory, size of the queue with the objective of minimizing execution time and therefore energy consumption. Our approach is based on a new approach PSO (Particle Swarm Optimization) to solve the problem of placement.**

Key words : **Keywords**: Mapping, Scheduling, PSO algorithm, Network-On-Chip, Multi-objective optimization, Dijkstra algorithm

I. Introduction

As silicon [6] technology keeps scaling, it is becoming technically feasible to integrate entire and complex systems on the same silicon die. As result of this Systems on Chip (SOC) are inherently heterogeneous and therefore complex, they are often formed multiple processors of different types (RISC, DSP; ASIC, for examples) features dedicated hardware or reconfiguration and peripheral.

To extend the integration and perform the design sweeps adopted the traditional concept of computer networking component interconnect-based routers, switches. well, a network on chip (NOC) or plus generally MPSoCs are a relatively aim new approach to integrated circuits on a platform SoC.[10]

Then MPSoCs and NoC are widely used in embedded systems (such as cellular phones, automotive control engines, etc..) where, once deployed in field, they always run the same set of applications.

In this paper we will focus on mesh-based NoC architectures, in which resources communicate with each other via mesh of switches that route and buffer messages. A ressource is generally any core : a general processor GP, a memory, an FPGA, DSP. A two dimensional mesh interconnection topology is simplest from a layout perspective and the local interconnection between resources and switches are independent of the size of the network.

Nevertheless, routing in a two dimensional mesh is easy, resulting in potentially small switches, high bandwidth, short clock cycles, and overall scalability [9]. One of the most onerous tasks in this context is the topological mapping of the resources on the mesh in such a way to optimize certain performances indexes (e.g power, performance). Mapping is, in fact, a problem of quadratic assignment that is known to be NP-hard.

The size space search of the problem increase exponentially with the system size depending on number of resources, tasks and communications. It is therefore of strategic importance to define methods to search a mapping that will optimize the desired performance indexes. In addition, the strategies have to handle a multi criteria exploration of the space of possible architectural mapping alternatives. The objectives to be optimized are, in fact, frequently multiple rather than single, and are almost always in contrast with each other. There is therefore non single solution to the problem of exploration (i.e single mapping) but a set of equivalent (i.e not dominated) possible architectural alternatives, featuring a different trade-off between the values of the objectives to be optimized (Pareto Set) [7].

Then a critical task for recent MPSoCs is the minimization of the energy consumed. We start from a well-characterized task graph, a directed acyclic graph representing a functional abstraction of the application that will run on the MPSoC. Each task is characterized by the number of clock cycles used for its execution. Clearly the duration of each task and the energy spent for running it depends on the clock frequency used during the task execution.

In addition, tasks connected by arcs in the task graph communicate if they are allocated to different processors.

Each edge is characterized by a number that corresponds to the total number of bytes exchanged between two tasks. Defining the optimal allocation, scheduling and voltage scaling for minimizing energy in MPSoCs is the aim of this paper. Energy is consumed during task execution and task communication.

The problem we face is very complex. Because we solve in the same time the allocation of tasks to the processors and find the optimal path allocation (or communication mapping) referred in literature as Network Assignment [8].

For this we have used two methods, one based on PSO algorithms and other on Dijkstra's algorithm. The solution must also verify some constraints such area, memory, load balancing, link speed, bandwidth and certainly hard real time constraints.

Our contribution differs from the above in that we use a new method in such research area and in the other hand we try to map tasks on processors and in the same time map optimally communications on the links of topology NoC.

The work presented in this paper is a contribution to solving a widespread problem in the field of system design, embedded the placement of a large application on an architecture (NOC). Our work in an application is represented by a set of tasks that communicate with each other by sending message via bus on a heterogeneous architecture.

Our role is to place the tiles (task) on different elements (core) of architecture with the objectives of minimizing time execution and the energy consumption under the constraints of load balancing, bandwidth, available memory and size of the queue waiting processors.

To solve this problem, we used in the context of our present work, a new meta-heuristic algorithm Particle Swarm. it has proved its effectiveness in many fields such as optimization of networks, image processing and even control of industrial systems but it was never applied in our domaine.

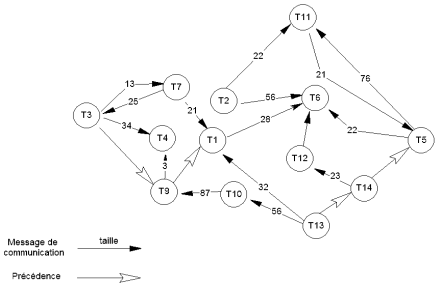

Fig. 1. Task graph

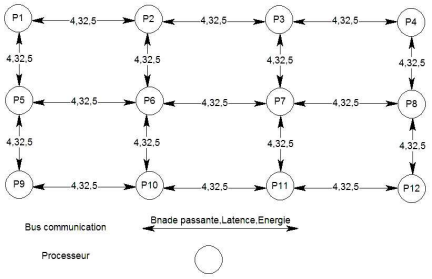

Fig. 2. NoC graph

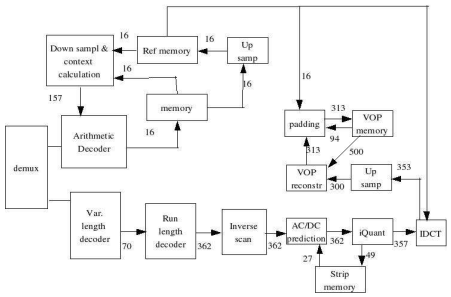

Fig. 3. block diagram of video object plane decoder with communication BW (in MB/s).

II. Problem definition and Formulation

The communication between the cores of SoC is represented by core graph.

Definition 1 : The Task Graph (CG)is a directed graph, $G(V, E)$ with each vertex $V_i \in V$ representing a task and the directed edge (v_i, v_j), denoted as $e_{ij} \in E$, representing the communication between the tasks v_i and v_j. The weight of the edge e_{ij}, denoted by Q_{ij} represents the bandwidth of the communication from v_i to v_j. The connectivity and link bandwidth of the NoC is represented by the NoC topology graph.

Definition 2 : The NOC Topology graph (NT) is a directed graph $P(U, F)$ with each vertex $u_i \in U$ representing a node in the topology and the directed edge (u_i,u_j) denoted as $f_{ij} \in F$representing a direct communication between the vertices's u_i and u_j. The weight of the edge f_{ij}, denoted by bw_{ij} represents the bandwidth available across the edge f_{ij} (see figure 1).

The mapping of the task graph $G(V, E)$ on to the processor graph $P(U, F)$ is defined by one to one mapping function map :

map : $V \rightarrow U$, s.t $map(v_i) = u_j \ \forall v_i \in V \ \exists u_j \in U$.

The mapping is defined when $\mid V \mid \leq \mid U \mid$ (see figure)

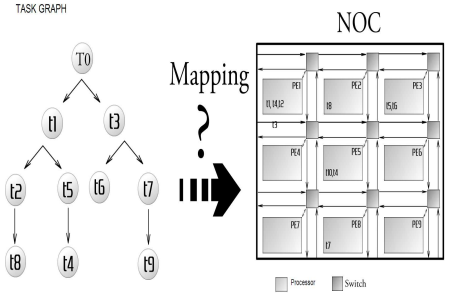

Fig. 4. Shows the mapping of task graph on NoC graph

A. Mathematical formulation

For an CG each node represent one task with its characteristics or property.

Let $T = t_1, t_2, ..., t_n$ be the set of all tasks represented by CG.

$P = p_1, p_2, ..., p_s$ is the set of processors represented by the nodes in NT.

We consider that each processor p can run in different modes m_1, m_2 or m_3.

We model the allocation problem with binary variables X_{ij}^m such that [6]:

$X_{ij}^m = 1$ if $task_i$ is mapped on the processor j and runs in mode m, o otherwise.

$d_{ij}^m =$ duration of execution task i on processor j running at mode m.

$d_{ij}^m = \frac{WCN_{ij}^m}{f_j^m}$ where WCN_{ij}^m is the number cycle needed by $task_i$ to be executed on processor j at mode m.

f_j^m is the frequency of the clock for processor j at mode m.

dl_i is deadline for $task_i$. The time at wish $task_i$ must be terminated.

There $dl_{final} = dl_n$ is deadline of last $task_n$ and of the application.

Q_{ij} is the size of data moved between $task_i$ and $task_j$.

dQ_{ijpq}^m is duration of communication between tasks i and j if they are assigned respectively to processor p and q at mode m (we see after how we compute dQ_{ijpq}^m).

q_{pq}^m is the duration of one unit (octet or bit) communication between p and q at mode m.

e_{pq}^m energy consumption for one unit from p to q at mode m.

If there is not a direct link between p and q let $\mu(p,q) = (pi, pj)$ be path form p to q.

Then duration using links is :

$dQ_{ijpq}^m = \sum Q_{ij} \times q_{plpk}^m$ where $(pl, pk) \in \mu(p,q)$ and $i \neq j$, $p \neq q$.
Let us note that if tasks i and j are mapped to the same processor the duration is negligible in comparison with case where they are mapped to different processors. therefore in the previous expression i differ of j and also p differ of q.
And consumption for communicating $task_i$ and $task_j$ if they are assigned to p and q at mode m towards the same path is :

$E_{ijpq}^m = \sum Q_{ij} \times e^m{}_{plpk}$ where $(pl, pk) \in \mu(p,q)$

Since we also assume take into account communication, we assume that two communicating tasks running on the same processor do not consume any energy and do not spend any time (indeed the communication time and energy spent are included in the execution time and energy), when if they are allocated on two different processors, they both consume energy and spend time. Each path contain some switch and router that need power consumption and duration.[5]
Ascia and al [7] address this problem but not explain how compute this. In This work we will consider an average value of consumption (Csw) and duration (dsw); we can estimate theses considering communications, input and output to router as stochastic.

Then if $\mid \mu(p,q) \mid$ is the length of path $\mu(p,q)$, the total consumption of switch and router on this path is :

$$\mid \mu(p,q) \mid +1) \times C_{sw}$$

And the total duration is :

$$\mid \mu(p,q) \mid +1) \times d_{sw}$$

We can now done equation of total consumption due to communication between $task_i$ and $task_j$ assigned respectively at processor p and q at mode m :

$$E_{ijpq}^m = E_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times C_{sw} \quad (1)$$

and duration due to communication as :

$$dQ_{ijpq}^m = dQ_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times d_{sw} \quad (2)$$

We explicit now total consumption and duration of processors.
For one $task_i$ mapped at $processor_p$ the duration is the sum of it's start time and duration of all its communications with other tasks mapped to other processors. The strategies scheduling is LS with ASAP (As Soon As Possible). Than for the $task_i$ mapped on $processor_p$ its duration D_{ip} is done by the following equation :

$$D_{ip} = d_{ip}^m + dQ_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times d_{sw} \text{with} i \neq j, p \neq q \quad (3)$$

$Dstart_i$ is the time at wish $task_i$ begin execution. it is equal at the time of the end of the last task which precedes it. The duration of the application mapped on many processors it is equal at time end of the last task. Let D be total duration including time execution and communication over links of all

tasks.

$$D = Dstart_n + D_n \quad (4)$$

Were D_n it is the duration of $task_n$ that is the last task. $Dstart_n$ is the time at wish $task_n$ begin execution.

The total consumption including processor, link and switch consumption is done by the flowed equation :

$$E_{pr} = \sum_{i=1}^{N} \sum_{p=i+1}^{s} \sum_{m=ml}^{mn} x_{ip}^{m} \times ET_{ip}^{m} \quad (5)$$

Where ET_{ip}^{m} is the consumption of $task_i$ if it is assigned on processor p at mode m.
Then E_{pr} is total energy consumed by all processors in NoC.

$$E_{com} = \sum_{i=1}^{N} \sum_{j=i+1}^{N} \sum_{p=1}^{s} \sum_{q=p+1}^{s} \sum_{m=ml}^{mn} x_{ip}^{m} \times x_{jp}^{m} \times E_{ijpq}^{m} \quad (6)$$

E_{com} is total energy consumed by network (links, routers or switch)to assure all communications between all tasks overall the NoC.

B. Our multi-objective Model

Our method is based on evolutionary computing method and an optimizer path. We have to search set of solutions under $multi - objective$. We consider here two objective total duration and consumption.

First objective is duration D (4)
The second objective is the total power consumption done by E such that :

$$E = E_{pr} + E_{com} \quad (7)$$

Note date during computing D and E we look for the shortest path between two cores which satisfy the constraints (such bandwidth and buffer). To obtain this we used dijkstra. we refer to this problem with AAS (Assignation Affectation and Scheduling.)

III. AAS Problem resolution

The rationale behind our approach is the minimization of total power consummation in the order to augment the autonomy of system Nevertheless try to reach this objective increase time computing. Or embedded applications are generally real time and of course the deadline must not be exceed. minimizing power consumption increase time computing and minimizing time computing increase power consumption. We have here two contradictory objectives, what returns this very complex problem. Multi-objective problems have a set of Pareto-optimal solutions. each solution represents a different optimal trade-off between the objectives and is said "non-dominated" since it is not possible to improve one criterion without worsening another. We propose a multi-objective approach based on Particle Swarm optimization technique to solve our AAS problem.

A. PSO: Particle Swarm Optimization

A.Capone and M.Cesana proposed [3] an evolutionary population-based heuristic for optimization problems. It models the dynamic movement or behavior of the particles in a search space. By sharing information across the environment over generations, the search process is accelerated and is more likely to visit potential optimal or near-optimal solutions. PSO has been extended to cope with multi-objective problems which mainly consist of determining a local best and global best position of a particle in order to obtain a front of optimal solutions. One of the welle-known multi-objective techniques based on PSO algorithm is MOPSO [4]. It is able to generate almost the best set of non dominated solutions close to the true Pareto front. The main algorithm is given below.

$$E_{ijpq}^{m} = E_{ijpq}^{m} + (\mid \mu(p,q) \mid +1) \times C_{sw} \quad (8)$$

Algorithm 1 MOPSO-Main

input : Swarm at iteration t S^t, MaxArchiveSize, MaxIteration
Output : Repository REP
$Step0$:Initialization of Swarm
Initialize S at iteration $t = 0$
for each i $\in S^0$ **do**

  for each dimension d **do**
    Initialize $position_i$, save $pBest_i$, initialize velocity
    Specify $lowerbound_i$ and $upperbound_i$
  end for
end for
$Step1$: Evaluation of particles S
$Step2$: Update REP
for each i $\in S^t$ **do**
  compVector(i,REP)search-insert(S,REP)
end for
$STEP3$: Generate Mapping(associative grid): makeCost(Mincost)
$Step4$: Update Swarm :
for each i $\in S^t$ **do**

  for each dimension d **do**
    $Update - velocity_i$, $Update_position_i$
  end for
end for
$Step5$: Boundary chek
$Step6$: Update pBest
$Step7$:
if $t > MaxIteration$ **then**
  Stop
end if
$t = t + 1$ and GO TO $Step1$

The following are the phases involved in the resolution of the proposed algorithm. In continuous optimization problems, getting the initial position and velocity is more straightforward because random initialization can be used. However, since the mapping problem is a constrained optimization problem, the initial positions must represent feasible solutions. Thus, they need to be designed carefully.[2]
A position in the search space represents a set of assignments that is a solution to the problem. In our case, each position provides information about how processor in the NoC will execute each task. Than, for each position in the swarm, we assign a Boolean value to the variables X_{ij}^{m}. We consider a feasible solution, a solution that satisfies all hard and soft constraints. During the search, only non-feasible solutions that violate some soft constraints can be included in the population. This increases the likelihood of a non-feasible solution to mutate and provide a feasible one in later generation.

B. Algorithm description

Given the rapidly changing and increasingly complex systems on chip (SoC - System on Chip) to systems on chip multi-processors (MPSoC - multiprocessor SoCs), interconnection of communication modules or cores (IP – Intellectual Property) constituting these systems, has undergone a change both in topology of the structure. This responds to the constraints of performance and cost related to the complexity and the increasing of interconnected modules or IPs. Currently this process is moving towards the integration of a communication network on chip, implementing the transmission of data packets to nodes interconnected network corresponding to modules or IPs (processors, memory, peripheral controllers connected, etc..). This transmission is done through routers forming the network and implementing rules of referral and routing packets across the network.
In the design flow of an embedded system, the stage of investment and is directly related to the implementation of the application on an architecture specialist. The entries in this phase are:

- An application model
- A model of target architecture
- Constraints of performance and energy
- The objective functions to optimize

The output of this phase is an allocation of tasks and communications in natural resources, according to various tasks on these resources.

To run a distributed application, it is necessary to determine the best placement of spots that compose the target NoC architecture while reducing the execution time and energy constrained load balancing, memory, and the size of the queue. Our proposal for solving the problem of mapping is defined as follows:
The particle is a representation of the solution of the problem in this case describes the investment. If you have a NoC Mesh M processors running at M modes and an application

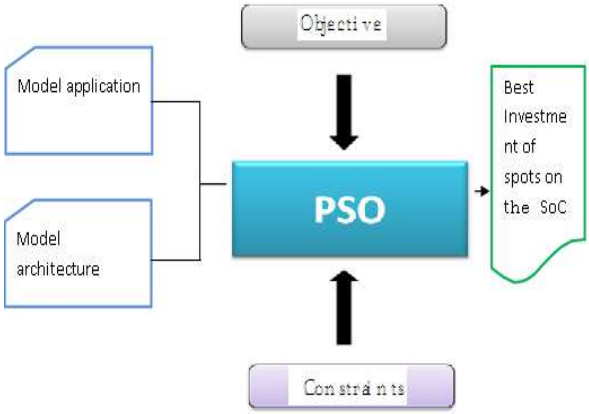

Fig. 5. PSO Algorithm

with N tasks when the particle is a matrix of N and S line column.
As an example take 2 processors and 3 tasks, then the particle is the mapping of 3 tasks: $Tasks2$ and 3 are assigned to P_1, $task1$ is assigned to P_2.

P_1	P_2
0	1
1	0
1	0

TABLE I
ONE PARTICLE

Get a set of points describing the Pareto front :

- Estimated front by iterative algorithms generate points near the front
- eliminating dominated points
- Problems of convergence:

1) approach the front
2) cover the entire front
3) Concept archive: Keep each iteration all the points not dominated.

Then The problem of placement of tasks in an application on a NoC to minimize cost objective functions. It can be formulated as follows:

- Given the graph of application (size of the task type of Soc, runtime memory required by SoC bandwidth required for a message and message size).
- Given the architectural graph (speed performance by mode, power consumption by mode, load balancing (load minimum and maximum), available memory in SoC size of the queue and bus latency and energy consumption due to transmission
- From the placement of tasks on the SoC's by different modes. This is equivalent to Minimize F (X): (f (time) f (energy)) The determination of the fitness function (or

adaptive function evaluation) involves several steps. Each time a swarm is generated according to the fitness of each particle must be evaluated.

- A particle represents a distribution of tasks of the application in the target NoC architecture.
- For a particle, move the communication costs of all messages for each message eliminating paths whose bandwidth is less than that required by the message (using the algorithm of the shortest path), then: We calculate the execution time of each task by mode which has been allocated within the SoC.

The Pareto Front is generated from existing solutions in the archive.

IV. Experimentations and results analysis

It's the first time who the PSO is applied in such area research. For now our first objective is to adapt this method to mapping real Time Applications on NoC by defining it's variables and parameters. We can study in the following works the performance of this algorithm in this area. The algorithm is coded in JAVA programming language and all the experiments were carried out on a Pentium 3.2 GHz. We have varied

]

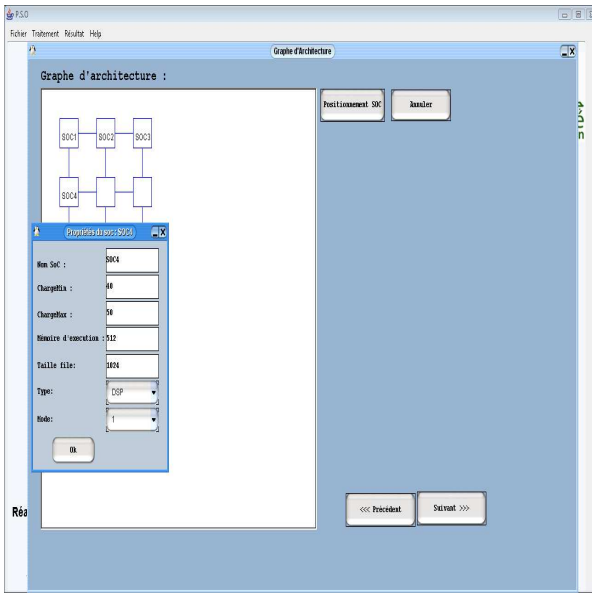

Fig. 6. Application interface

nombre of generations for a same exemple to know wath is the best interval of generations for similaire size of applications. In general study the searchers fixe this nombre at 20 and our experiments shows that not necessarily to take this nombre bigger than 20 nevertheless our exemple is not taken with a big size(see figure 7).

The other parameter of the algorithm is the size of swarm. It is also important because it influences the search convergence. Our experiment show that is important to take Swarm size near 100.(see figure 8)

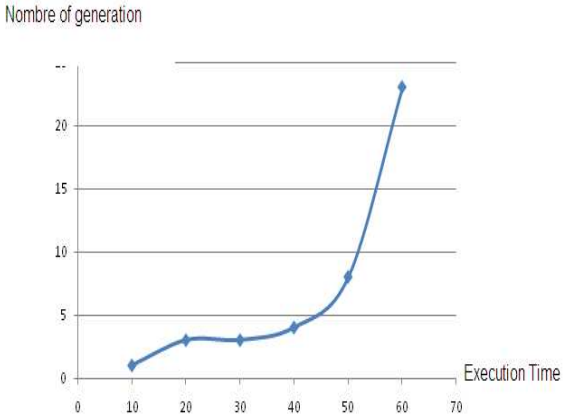

Fig. 7. influence of nombre generation on Time computing

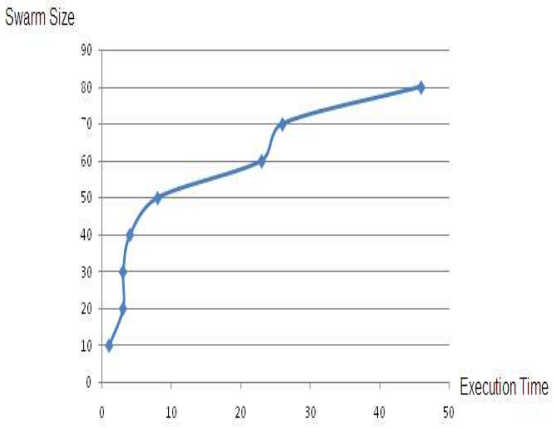

Fig. 8. influence of size swarm on Time computing

Figure (9) shows the results of different experiments for applications of average size that we have obtained. The achieved results and performances indicate that the proposed algorithm has a polynomial complexity and that it is adapted for computationally hard problems.

V. Conclusion

In this paper an algorithm for allocation and scheduling has been proposed exploiting the method of PSO and Disjkstra's shortest path algorithm. Under that we can't assert that this algorithm gives exact results for all kinds of problems. The proposed framework searches a good mapping for heterogeneous distributed embedded systems and of course this approach can easily be applied to regular architectures and first experimental results show that the global time for research is reasonable. Nevertheless for asserting that this method can be used for solving problems with large sizes we must experiment the algorithm using examples with many hundreds of PE and tasks. Then we consider for now this work a first step in our research for a good meta-heuristic which can be crossed with other exact method to solve th global problem known as GILR.

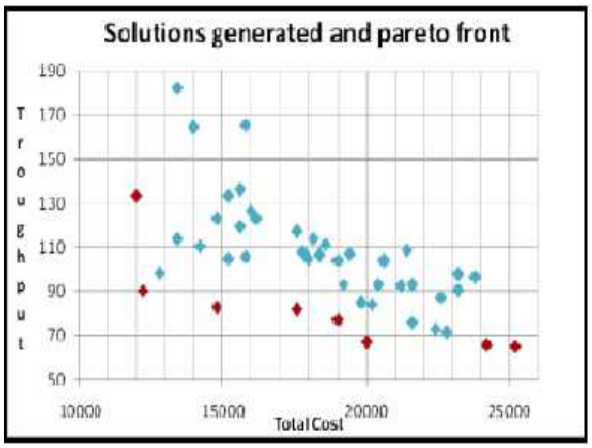

Fig. 9. Feasible solutions generated with the same initial parameters

This will be the continuation for this work and our perspective.

REFERENCES

[1] IR.Quadri, P.Boulet, S.Meftali and JL.Dekeyser, *Using an MDE Approach for Modeling of Interconnection Networks*, Proceedings of the The International Symposium on Parallel Architectures, Algorithms, and Networks. ISPAN, pp 289-294, 2008.

[2] HJ.Escalante, M.Montes and LE.Sucar *Particle Swarm Model Selection*, The Journal of Machine Learning Research. pp.405-440, vol 10, 2009.

[3] A.Capone, M.Cesana, I.Filippini and F.Malucelli, *Optimization models and methods for planning wireless mesh networks* , Computer Network. accepted for publication January 2008.

[4] TDL.Truong, *Shared protection for multi-domain networks*, Ph.D.thesis. University of Montreal. Sept.2007

[5] AH.Benyamina, P.Boulet and B.Beldjilali , *An Hybrid algorithm for Mapping on NoC Architectures*, META'08 . Hammamat,Tunisia,October, 2008.

[6] A.Benymina and P.Boulet,*Multi-objective Mapping for NoC Architectures* , Journal of Digital Information Management (JDIM). ,pages 378-384, volume 5, number 6, 2007.

[7] G.Ascia, V.Catania and M.Palesi, *A Multi-objective Genetic Approach to Mapping Problem on Network-on-Chip* Journal of Universal Computer Science. , pages 370-394, vol 2, (2006)

[8] D.Shin, J.Kim,*Power-Aware Communication Optimisation for Networks-on-Chips with Voltage Scalable Links* , in Proc. CODES+ISSS'04, , pp.170-175, Sep.2004.

[9] L.Benini, D.Bertozzi, A.Guerri and M.Milano, *Allocation, Scheduling and Voltage Scaling on Energy Aware MPSoCs*, CPAIOR pp.44-58, 2006.

[10] T.Bjerregaard and S.Mahadevan, *survey of research and pratices of network-on-chip*, ACM Comput Surv. 38(1):1, 2006.