Static Scheduling of a Time-Triggered Network-On-Chip based on Memetic Algorithm

Heyuan Shi
School of Software
Tsinghua University
Beijing, China
Email: shy15@tsinghua.edu.cn

***Abstract*—Time-Triggered Network-on-Chip (TTNoC) is a time-triggered system deployed on the Network-on-Chip (NoC). It ensures the reliable and safety-critical communication for modern embedded multiprocessor systems. To guarantee the real-time requirements, each message for communication is transmitted by a predefined schedule. However, synthesizing the scheduling is a challenging problem because both path contention and the temporal constrains should be considered. This paper introduces a memetic algorithm (MA) integrating local search into the genetic algorithm, to solve this problem. The experimental results show that the better performance of our memetic algorithm in compare with the general genetic algorithm.**

I. Introduction

Time-triggered systems is one of the real-time systems with high throughput and temporal predictability for multiprocessor systems. Since isolating the inherent fault as well as satisfying the temporal predictability of communication, time-triggered systems is widely deployed in various field, e.g. TTEthernet integrates time-triggered concept into the general Ethernet [1], which is adopted as a new standard of the avionic network [2]. TTP and FlexRay adopted in aerospace industry [3].

TTNoC integrates the time-triggered concepts into the Network-on-Chip (NoC) for safety-critical real-time embedded systems which desire the tight and predictable communication latency. In practical, GENESYS is proposed as a cross-domain embedded system architecture, which influenced by the concepts of and the experience with the time-triggered architecture [4].

There are two main architectures of time-triggered system, bus-based and network-based architecture [5]. Bus-based architecture just enable there is only one message transmitted on the system at the same time. Therefore each message own its unique duration to occupy the physical link for communication. Unlike the bus-based architecture, The network-based architecture allows a set of message to be transmitted at the same time, on condition that the physical links the messages used are non-overlapping. It entails messages sharing the time domain to communicate. Fig. 1 shows the differences between bus-based and network-based time-triggered architecture. In bus-based architecture, all the messages (m_0,m_1,m_2) achieve temporal separation. In network-based architecture, the set of messages (m_0,m_1,m_2) is able to share the time domain if there is no spacing conflict among the transmission of them.

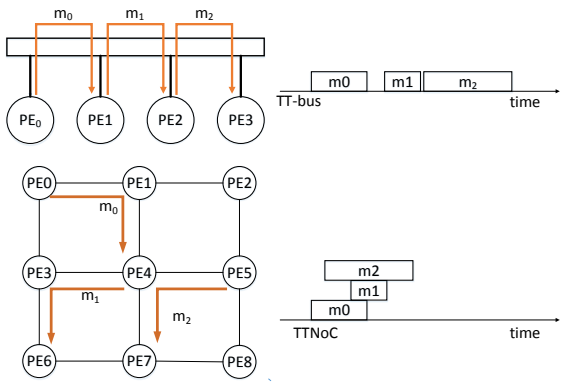

Fig. 1. Difference between bus- and network-based time-triggered architecture

The TTNoC is network-based, therefore the scheduling of TTNoC is to determine the duration of each message on the TTNoC and ensuring there is no contention with the communication. Static scheduling is adopted to synthesize the scheduling which resolves contention, avoiding dynamic arbitration of intercommunicate resources distribution. So that the communication on the TTNoC is predefined while satisfying the scheduling constrains. However, synthesizing such a scheduling is rather complicated which is known as NP-complete [5], [6].

The main contribution of this paper is presenting a static scheduling approach to resolve the TTNoC scheduling problem by the Memetic Algorithm (MA). MA is widely used as a synergy of evolutionary or any population-based approach with separate individual learning or local improvement procedures for solution search [7].

The rest of this paper is organized as follows. We first review the related work in the TTNoC scheduling in Section II. The system model is given in Section III, followed by the schedule constrains in Section IV. Next we presents the problem formulation and assumptions in Section V. Then the memetic algorithm is described in Section VI. The experiments as well as synthesis results are presented in Section VII. Finally, we conclude in Section VIII.

II. Related Work

There are sufficient literatures about the scheduling of time-triggered network, which is similar to the scheduling

of TTNoC. [6] consider the scheduling problem for time-triggered multi-hop networks. A pure satisfiability modulo theories (SMT) formulation is presented followed by an incremental method to improve the scalability. [8], [9] formulate the schedule problem using first-order logical constraints and present alternative methods to find a solution based on SMT and mixed integer programming (MIP) solvers. In [10] the authors presents a decomposition approach based on SMT solver for extremely large time-triggered network.

The scheduling problem also appear in the various domain. [11] resolve the scheduling problem on the wireless network by SMT solver. [12] formulate the co-synthesis problem of task and communication schedules as a Mixed Integer Programming (MIP) model taking into account a number of Ethernet-specific timing parameters. Holistic Scheduling in Time-Triggered In-Vehicle Networks was studied in [13]

The scheduling problem of TTNoC has been studied firstly in [5] which is integrates SMT solver into classical heuristic algorithm. [14] introduces an optimal scheduler based on a Boolean SAT solver for a TTNoC. [15] introduces a scheduling model based on Mixed Integer Linear Programming (MILP) combining both time-triggered and event-triggered messages on NoC. In [16], A heuristic algorithm for scheduling on the scalable communication structure like a NoC is presented.

III. System Model

In this section we presents the basis and model of TTNoC. The system architecture we adopted is shown in [17].

A. Architecture Model

TTNoC is a system with a set of real-time tasks/applications mapped on a multi-core interconnection platform. It composes of a set of identical processing elements interconnected by physical links and the switches on the TTNoC. The switches are connected by bidirectional physical links and Each link connect a pair of switches. The topologies of a TTNoC can be either regular or irregular.

The architecture of a TTNoC is modeled by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$. The set of vertexes $\mathcal{V} = \{v_1, \ldots, v_m\}$ represents the set of m *identical* communication nodes (the processing elements and switches) while the edges $\mathcal{L} \subseteq \mathcal{V} \times \mathcal{V}$ comprises the communication physical links *directly* connecting the nodes. The physical links are full-duplex, allowing thus communication in both directions. Therefore, given $v_i, v_j \in \mathcal{V}$, $\langle v_i, v_j \rangle \in \mathcal{L}$ implies $\langle v_j, v_i \rangle \in \mathcal{L}$, where $\langle v_i, v_j \rangle$ is a pair denoting a directed physical link connecting two adjacent nodes, from a source node v_i to a sink node v_j.

The TTNoC composed of a set of identical process elements and physical links. Since the width of physical links as well as the performance of switches and links is the same, the properties of each hop can be regarded as system properties of TTNoC. Therefore TTNoC $\mathcal{G}$ is equipped with a set of properties. We use $W_{\mathcal{G}}$ to denote the width of physical links in the TTNoC, $SD_{\mathcal{G}}$ for the switching delay in the switches, $HD_{\mathcal{G}}$ for the link delay per hop which means propagation cost between adjacent nodes, and $MT_{\mathcal{G}}$, described in the scheduling model, for the macrotick of network representing the time-line granularity of the physical link.

Given two nodes $v, v' \in \mathcal{V}$, a *route* from v to v', denoted by $r^{\langle v, v' \rangle}$, is defined as a sequence of the form $\langle v_{k_0}, v_{k_1} \rangle \langle v_{k_1}, v_{k_2} \rangle \ldots \langle v_{k_{i-1}}, v_{k_i} \rangle \langle v_{k_i}, v_{k_{i+1}} \rangle \ldots \langle v_{k_{n-1}}, v_{k_n} \rangle$ satisfying $v_{k_0} = v$, $v_{k_n} = v'$ and $(\forall i \in [1, n]) \langle v_{k_{i-1}}, v_{k_i} \rangle \in \mathcal{L}$. The length n of the route is written as $|r^{\langle v, v' \rangle}|$. A route $r^{\langle v_i, v_j \rangle}$ represents a *possibly undirected* physical link from v_i to v_j in the TTNoC, hence its length $|r^{\langle v_i, v_j \rangle}|$ is the number of hops along this link.

B. Message Model

The communication of process elements on the TTNoC is based on messages. A message consists of header and data. Header includes the information for message forwarding on the TTNoC, e.g. address of source and destination. A flit is the unit of a message that can be transmitted over the TTNoC. Only periodic messages are discussed in this paper because it is the typical case in real-time embedded systems. But in practice, the fixed time slot can be preserved in terms of the route and deadline periodically for sporadic messages.

We denote the set of n time-triggered messages on the TTNoC by $\Gamma = \{\tau_1, \ldots, \tau_n\}$. Each message τ_i is modeled by the tuple $\langle \tau_i.H, \tau_i.S, \tau_i.T, \tau_i.D \rangle$, where $\tau_i.H$ denotes the header of the message in flits, $\tau_i.S$ denotes the size of the message in flits, $\tau_i.T$ and $\tau_i.D$ are the period and the relative deadline of the message, respectively.

C. Scheduling Model

When a communication happens on the TTNoC between two nodes, i.e. a source node $v_i \in \mathcal{V}$ sends a message τ_k to a sink node $v_j \in \mathcal{V}$ on the TTNoC, the message τ_k is transmitted on some route $r^{\langle v_i, v_j \rangle}$. And the communication nodes along the route $r^{\langle v_i, v_j \rangle}$ to forward the message.

The TDMA scheme is deployed for the scheduling of communication. And the time granularity is *macrotick* that the periods and phases of messages directly match to this time format.

Based on the architecture and message model, we can model the scheduling of communication as a set $\mathcal{S} = \{s_1, \ldots, s_n\}$, in which each s_i denotes the communication scheduling of the message $\tau_i \in \Gamma$. Each communication $s_i \in \mathcal{S}$ is defined using a tuple $\langle s_i.R, s_i.\phi, s_i.T, s_i.D, s_i.L \rangle$, where $s_i.R$ is the route along which the message τ_i is forwarded, $s_i.\phi$ is the offset of the communication, $s_i.T$ and $s_i.D$ denote, respectively, the period and the relative deadline of the communication, and $s_i.L$ is the duration of the communication.

For the whole set $\mathcal{S}$, the sets of the route, the period, the relative deadline and the delay of each $s_i \in \mathcal{S}$ are defined as expected, denoted by $\mathcal{S}.\mathcal{R}$, $\mathcal{S}.\Phi$, $\mathcal{S}.\mathcal{T}$, $\mathcal{S}.\mathcal{D}$ and $\mathcal{S}.\mathcal{L}$, respectively.

Because of the macrotick as the granularity of communication, for each communication s_i, we have the following formulas:

$$s_i.T = \lceil \frac{\tau_i.T}{MT_{\mathcal{G}}} \rceil \tag{1}$$

$$s_i.D = \lceil \frac{\tau_i.D}{MT_{\mathcal{G}}} \rceil \tag{2}$$

$$s_i.L = \lceil \frac{|s_i.R| \times (SD_{\mathcal{G}} + HD_{\mathcal{G}}) \times \lceil \frac{\tau_i.S + \tau_i.H}{\tau_i.H} \rceil}{MT_{\mathcal{G}}} \rceil \tag{3}$$

where $MT_{\mathcal{G}}$ denotes the value of macrotick, $SD_{\mathcal{G}} + HD_{\mathcal{G}}$ is the delay per hop in the forwarding, that $SD_{\mathcal{G}}$ is the delay on the physical link and $HD_{\mathcal{G}}$is the switching delay.

The formula is based on [18] with the transformation of adopting macrotick as the unit. Given an $s_i \in \mathcal{S}$ and its route $s_i.R$, $s_i.T$, $s_i.D$ and $s_i.L$ can be derived by formulas 1, 2 and 3. Formulas 1 and 2 denote, respectively, the period and the deadline of the communication s_i in macrotick. Formula 3 computes the time cost in macrotick, from the first flit sent by the source node to the last flit received by the sink node, according to store-and-forward (SAF) switching scheme [18]. Moreover, other scheme of switching on the NoC can be also deployed in the TTNoC according to the requirements. In this case, the delay formula 3 may be changed according to the given switching scheme.

As a result, to synthesize the set of communication $\mathcal{S}$ is to calculate the offset $s_i.\phi$ for each communication $s_i \in \mathcal{S}$. Therefore, the key to scheduling of TTNoC is to derive $\mathcal{S}.\Phi = \{s_i.\phi \mid s_i \in \mathcal{S}\}$ based on the given architecture model $\mathcal{G}$ and the messages model Γ with *scheduling constrains*.

IV. SCHEDULING CONSTRAINS

The scheduling constrains for general time-triggered multi-hop network is presented in [6]. The scheduling constrains on TTNoC is similar with general time-triggered network, while some differences existing in terms of our model.

Scheduling of communication $\mathcal{S}$ should resolve the constrains on the TTNoC. The synthesized scheduling must satisfy the separation among the set of communication in time domain or space domain, which mainly include offset constrains and link constrains.

In our TTNoC, scheme of bufferless NoCs is adopted that no buffering of messages takes place. It lead to significantly less area and power consumption for TTNoC [19]. The route of each message is located on its header. The switches just simply forward the message according to the address of next-hop switch, without concerning the communication schedule. Therefore, unlike the general time-triggered network, the switch memory constrains are not included. Moreover for the simplicity of evaluation, the constraints of simultaneous relay, application-level, protocol-based and domain-specific is neither included.

A. Message Offset Constrains

For a scheduling of message $s_i \in \mathcal{S}$, the offset $s_i.\phi$ must be positive values and should guarantee that the message transmission is completed before the deadline of the communication. Therefore we have that

$$(s_i.\phi \geq 0) \cap (s_i.\phi + s_i.L \leq s_i.D) \tag{4}$$

B. Link Contention Constrains

For each communication, the message occupies the link in terms of routing strategy without contention, until all the message is received by sink node. The scheduling $\mathcal{S}$ should guarantee that no two messages that are both transmitted on the identical link time at any time. The synthesized scheduling must guarantee that the links on TTNoC is contention-free.

Therefore for any two of the scheduling of message $s_i, s_j \in \mathcal{S}, s_i \neq s_j$, we define $overlap(s_i, s_j)$ denotes whether there is common routing link between s_i and s_j or not. $overlap(s_i, s_j)$ equals 1 if two communications sharing routing link for at least one hop. Therefore we have that

$$overlap(s_i, s_j) = \begin{cases} 0 & s_i.R \cap s_j.R = \emptyset \\ 1 & s_i.R \cap s_j.R \neq \emptyset \end{cases} \tag{5}$$

The duration of scheduling s_i for message τ_i in its kth period is defined by $s_i(k)$, which equals

$$s_i(k) = [k \times s_i.T + s_i.\phi, k \times s_i.T + s_i.\phi + s_i.L] \tag{6}$$

Therefore for any $s_i, s_j \in \mathcal{S}, s_i \neq s_j$, we have that

$$s_i.R \cap s_j.R \neq \emptyset \Longrightarrow s_i(p) \cap s_j(q) = \emptyset \tag{7}$$

where $p, q \geq 0$, $p, q \in \mathcal{Z}$.

V. PROBLEM FORMULATION

The problem we are addressing in this paper can be formulated as follow. Given:

- The architecture of TTNoC $\mathcal{G} = (\mathcal{V}, \mathcal{L})$ consists of the set of communicating nodes $\mathcal{V} = \{v_1, \ldots, v_m\}$ and interconnection links $\mathcal{L} = \{\langle v_i, v_j \rangle \mid v_i, v_j \in \mathcal{V}\}$.
- The mapping of message to communication node $\{< \tau_i, v_j > \mid \tau_i \in \Gamma, v_j \in \mathcal{V}\}$. Each message $\tau_i \in \Gamma$ is allocated to a specific process element(communication node) $v_i \in \mathcal{V}$, thus the source node v_a and sink node v_b for each $\tau \in \Gamma$ are determined. It should be noted that the scheduling of mapping can be synthesized by genetic algorithm [20]. In our experiment the mapping is predefined.
- The set of tt-messages $\Gamma = \{\tau_1, \ldots, \tau_n\}$ with $\langle \tau_i.H, \tau_i.S, \tau_i.T, \tau_i.D \rangle$ for each $\tau \in \Gamma$.
- The route $s_i.R$ for each scheduling $s_i \in \mathcal{S}$ of message $\tau \in \Gamma$.

We aim to find the offset $\mathcal{S}.\Phi$ for messages Γ on TTNoC $\mathcal{G} = (\mathcal{V}, \mathcal{L})$ which satisfy the scheduling constrains.

We introduce the memetic algorithm to synthesize the set of offset $\mathcal{S}.\Phi$ such that satisfies the scheduling constrains. If it is impossible to synthesize a set of scheduled communication, then we try to minimize the amount of infeasible messages which violate the scheduling constrains.

VI. MEMETIC ALGORITHM

Memetic algorithm(MA) is a population-based hybrid genetic algorithm(GA) coupled with an individual learning procedure capable of performing local refinements[21].

Our memetic algorithm consists of two parts. We deploy the genetic algorithm as global searching for the set of

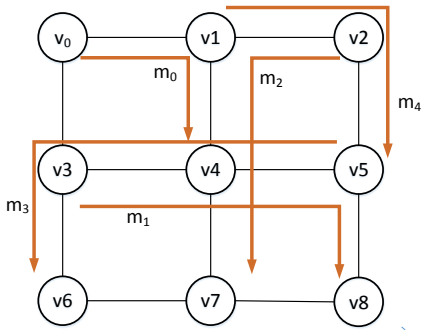

Fig. 2. An example of communication on the TTNoC

TABLE I
AN EXAMPLE OF SCHEDULING OF MESSAGES ON THE TTNOC

Scheduling	Route	Period	Delay	Possible Offset
s_0	$r^{(0,4)}$	2	1	$\{0,1\}$
s_1	$r^{(3,8)}$	4	1	$\{0,1,2,3\}$
s_2	$r^{(2,7)}$	4	1	$\{0,1,2,3\}$
s_3	$r^{(5,6)}$	8	2	$\{0,1,2,3,4,5,6\}$
s_4	$r^{(1,5)}$	8	1	$\{0,1,2,3,4,5,6,7\}$

communication to be scheduled. And choosing the subset of infeasible communications as the element to local search.

The MA requires genetic representation, fitness function as well as evolution strategy, i.e. crossover, mutation and local search. We firstly give an example of communication scheduling to explain the genetic representation. Next the pseudocode is shown. And then the detail will be discussed.

A. An Example of Scheduling on TTNoC

To show our memetic algorithm, we firstly give an example of scheduling $\mathcal{S}$ of messages Γ on a 3×3 mesh NoC, shown in Fig 2. There are a set of message to be scheduled on the TTNoC, we derive the scheduling model employing the architecture and message model, shown in TABLE I. Each scheduling $s_i \in \mathcal{S}$ own its period, duration as well as link in terms of the given route. It is noted that the period must be a positive power of two in terms of macrotick according to the timing specification of TTNoC [5].

We assume that the relative deadline is equal to period for each message for simplicity. The set of possible offset is denoted as $s_i.\Phi$ for $s_i \in \mathcal{S}$. $s_i.\Phi$ is derived by $0 \leq s_i.\phi \leq s_i.T - s_i.L$ according to the message offset constrains. The $s_i.\phi$ is generated among the $s_i.\Phi$. The set of possible offset $s_i.\Phi$ for the messages on Fig 2 is also shown in TABLE I.

B. Genetic Representation

The genetic algorithm employs global search consists of gene, chromosome, individual and population. The population consists of a set of individual and each contains a chromosome which represents a solution to the given question. A chromosome composes of several gene, which is the unit of chromosome. We define them on the scheduling on the TTNoC.

TABLE II
AN EXAMPLE OF POPULATION WITH 2 INDIVIDUAL

Gene	0	1	2	3	4	5	6	7
$indi0$	m_0, m_1		m_2		m_3			m_4
$indi1$	m_0	m_1		m_2	m_3		m_4	

In our memetic algorithm, we adopt hyperperiod $H(\mathcal{S}) = LCM\{\mathcal{S}.\mathcal{T}\}$ as the chromosome definition. Hyperperiod is the least common multiple of the offset among scheduling for messages. And the macrotick is defined as the unit of hyperperiod which is called gene. The length of chromosome equals the value of $H(\mathcal{S})$, e.g. the hyperperiod is 8 for the scheduling set on TABLE I.

The location of a gene on the chromosome denotes the offset $s_i.\phi$ of a message $\tau_i \in \Gamma$. The offset of scheduling is distributed on a gene. Therefore each chromosome on different individual represents a possible allocation of the offset set $\mathcal{S}.\Phi$. It should be noted that a single gene may consists multiple offset of messages, since several communication is able to offset at same time based on the TTNoC architecture if the TTNoC is out of contention.

A set of individual with respective chromosome composes the population. The size of population equals the number of individual. An example of population with two individual, based on TABLE I, is shown in TABLE II. And the example of chromosome and gene is also depicted.

C. Algorithm Description

According to the genetic representation, we formal give the algorithm process. The pseudocode is given as follow.

The memetic algorithm begins with an initial population Pop. The size of population $scale$ as well as the maximum iterations $maxIteration$ is predefined. The initial population is synthesized randomly. For a scheduling $s_i \in \mathcal{S}$, the offset $s_i.\phi$ in the initial individual is located randomly among the possible offset $s_i.\Phi$. Therefore the initial population satisfies the offset constrains. The algorithm is iterative executed until a feasible scheduling is found or reaching the given maximum iteration.

When the memetic algorithm get started, firstly the initial population is marked by the fitness function. The score evaluating chromosome of the individual is assigned to individual. The less the number of score, the better the individual. Therefore the individual with score equals zero represents a feasible scheduling is synthesized. The detail of fitness function is discussed in subsection VI-D.

Next, the operation of genetic algorithm, i.e. crossover and mutation, is deployed as global search. And after that the local search is adopted to the population if there are no feasible scheduling(individual with score of 0). The detail of search strategy deployed in global as well as local search, is discussed in VI-E and VI-F, respectively.

After the global and local search, each individual in the population is evaluated by fitness function again. And the new population for the next iteration is generated by selection according to the score of each individual. The individual with

Algorithm 1 Memetic Algorithm

Input:
the size of population $scale$
maximum iterations $maxIteration$
Output:
Best scheduling $\mathcal{S}$

for $i = 0$ to $maxIteration - 1$ **do**
  if $i = 0$ **then**
    /*mark the initial population*/
    generate the initial population $initialPop$
    mark($initialPop$)
    $\mathcal{S}$ = individual with least score **in** $initialPop$
    $Pop = initialPop$
  else
    /*global search*/
    crossover(Pop)
    mutation(Pop)
    mark(Pop)
  end if
  /*choose the individual best so far after global search*/
  for all individual $indi$ **in** Pop **do**
    if $indi.score < \mathcal{S}.score$ **then**
      $\mathcal{S} = indi$
    end if
    if $\mathcal{S}.score == 0$ **then**
      break
    end if
  end for
  localSearch(Pop)
  mark(Pop)
  /*choose the individual best so far after local search*/
  for all individual $indi$ **in** Pop **do**
    if $indi.score < \mathcal{S}.score$ **then**
      $\mathcal{S} = indi$
    end if
    if $\mathcal{S}.score == 0$ **then**
      break
    end if
  end for
  select(Pop, $scale$)
end for
return $\mathcal{S}$

less score own more possibility to reserve for the next population. The selection function ensure the size of population is equal to the value of $scale$ in each iteration.

D. Fitness function

To adopt the fitness function to evaluate individuals, we firstly transform chromosome to represent the duration of communication in a hyperperiod. Next we derive the link contention constrains among the scheduling according to the routing strategy, as the basis of our fitness function. Then the score of each individual is able to employed by fitness function.

TABLE III
AN EXAMPLE OF TRANSFORM OF CHROMOSOME TO REPRESENT DURATION OF MESSAGES

Duration	0	1	2	3	4	5	6	7
$indi0$	m_0 m_1		m_0 m_2		m_0 m_1 m_3	m_3	m_0 m_2	m_4
$indi1$	m_0	m_1	m_0	m_2	m_0 m_3	m_1 m_3	m_0 m_4	m_2

TABLE IV
AN EXAMPLE OF A ROUTING AND OVERLAP SET

Communication	Routing Link	Overlap Set
s_0	$\langle 0,1\rangle\langle 1,4\rangle$	s_2, s_4
s_1	$\langle 3,4\rangle\langle 4,5\rangle\langle 5,8\rangle$	s_4
s_2	$\langle 2,1\rangle\langle 1,4\rangle\langle 4,7\rangle$	s_0, s_4
s_3	$\langle 5,4\rangle\langle 4,3\rangle\langle 3,6\rangle$	
s_4	$\langle 1,4\rangle\langle 4,5\rangle$	s_0, s_1, s_2

For each scheduling $s_i \in \mathcal{S}$ for the messages on the TTNoC, The delay of scheduling s_i in kth period $s_i(k)$, is defined by formula 6. And we have that

$$s_i.H = \{s_i(k) \mid k \in [0, H(\mathcal{S})/s_i.T - 1]\} \tag{8}$$

where $s_i.H$ denotes the duration of a scheduling in hyperperiod. We define $\mathcal{S}.\mathcal{H} = \{s_i.H \mid s_i \in \mathcal{S}\}$ denotes the set of duration for each message scheduling in hyperperiod. Therefore, the complete duration in hyperperiod for each scheduling $\mathcal{S}.\mathcal{H}$ is determined. TABLE III is an example of the complete duration in a hyperperiod, based on two individuals in TABLE II. The scheduling model of messages is shown in TABLE I.

The link contention constrains should be derived after the transformation of individual. According to the predefined routing strategy, e.g. XY routing [18], the route $s_i.R$ of each scheduling of message $s_i \in \mathcal{S}$ is derived. Therefore we have the forwarding path per hop. By checking the routing of each communication by formula 5, for each s_i, the set of scheduling which is overlapped with s_i is derived. We denote the set of overlapped scheduling $O(s_i)$ for each s_i, and we have that

$$O(s_i) = \{s_j \mid overlap(s_i, s_j) = 1, s_i, s_j \in \mathcal{S}\} \tag{9}$$

Table IV shows the route based on XY routing and overlapped scheduling for the scheduling depicted in TABLE I. Since the $overlap(s_0, s_1) = 0$, $overlap(s_0, s_2) = 1$, $overlap(s_0, s_3) = 0$ and $overlap(s_0, s_4) = 1$, thus overlap set $O(s_0)$ is $\{s_2, s_4\}$.

For $s_i, s_j \in \mathcal{S}, s_i \neq s_j$, if s_i and s_j are overlapped, the conflict times between them in chromosome are derived to evaluate the individual. For each $s_i \in \mathcal{S}\}$, we define $C(s_i)$ to derive the set of scheduling conflicted with s_i. Therefore we have that

$$C(s_i) = \{s_j \mid s_i.H \cap s_j.H \neq \emptyset, s_j \in O(s_i)\} \tag{10}$$

A conflict function is defined to count the conflict times between s_i and s_j. We have that

$$conflict(s_i, s_j) = sizeof\{s_i.H \cap s_j.H\} \tag{11}$$

TABLE V
AN EXAMPLE OF MARKING TWO INDIVIDUAL BY FITNESS FUNCTION

Individual	Conflict	Conflict Times	Score
individual 0	s_0, s_2	$times(s_0) = 2, times(2) = 2$	4
individual 1	s_0, s_4	$times(s_0) = 1, times(4) = 1$	2

We define the total conflict times to s_i as $C(s_i).value$. And we have that

$$times(s_i) = \sum_{s_j \in C(s_i)} conflict(s_i, s_j) \tag{12}$$

If there is no conflict scheduling with s_i on the its duration, $C(s_i) \in \emptyset$. We define $T(\mathcal{S}) = \{s_i \mid C(s_i) \notin \emptyset\}$ to denote the set of conflict scheduling in a chromosome.

According to $T(\mathcal{S})$ the conflict times among any two of scheduling of message on the chromosome is determined. The fitness function defined to give each individual a score when executing $mark(Pop)$ in line 5, 12, 24 of the memetic algorithm. The output of fitness function is total conflict times in $T(\mathcal{S})$. Therefore for a individual $indi$, we have that

$$fitness(indi) = \sum_{s_i \in T(\mathcal{S})} times(s_i) \tag{13}$$

The score is assigned to the individual after fitness function. Since the score represents the conflict times, the less the score the better the individual. e.g. The the score of two individual in Fig II is shown in TABLE V.

E. Global search strategy

The general operation in genetic algorithm, i.e. crossover and mutation, is used in the step of global search.

For operation of crossover, we employ two of individual $indi_i, indi_j \in Pop$ among the population the parent to join the crossover in terms of the score. The lower scores of the individual, the higher possibility to be selected as parent. The crossover generates a new individual and its chromosome depends on the parent. The individual of parent with lower score own the higher possibility to determine the offset location on the chromosome. For the parent individual $indi_i$ and $indi_j$, we have the function

$$possibility(indi_i) = 1 - \frac{indi_i}{indi_i + indi_j} \tag{14}$$

to derive the possibility for each individual. Table VI shows an process of crossover. The $indi2$ is generated by crossover based on the two individuals in Fig II as parents. Because the score of $indi0$ is 4 while $indi1$ is 2, $indi0$ owns 67% while $indi0$have 33% of possibility to decide the offset when locating each scheduling $s_i \in \mathcal{S}$. The chromosome of $indi2$ consists of the offset of m_0, m_2, m_3 from $indi1$ and m_1, m_4 from $indi0$.

The operation of mutation random select an offset $s_i.\phi$ of scheduling $s_i \in \mathcal{S}$. Then selecting a new location of its offset randomly. TABLE VII is an example of mutation. It is noted that the probability of mutation for each individual in our implement is 50%.

TABLE VI
AN EXAMPLE OF CROSSOVER EMPLOYED INDIVIDUALS IN FIG II AS PARENTS

Parents	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi0$	m_0, m_1		m_2		m_3			m_4
$indi1$	m_0	m_1		m_2	m_3		m_4	
Child	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi2$	m_0, m_1			m_2	m_3			m_4

TABLE VII
AN EXAMPLE OF MUTATION

Before	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi0$	m_0, m_1		m_2		m_3			m_4
After	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi3$	m_0, m_1	m_4	m_2		m_3			

TABLE VIII
AN EXAMPLE OF LOCAL SEARCH OF m_0 AS ELEMENT

Before	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi0$	m_0, m_1		m_2		m_3			m_4
Score	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
4	m_0, m_1		m_2		m_3			m_4
2	m_1	m_0	m_2		m_3			m_4
After	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
$indi0$	m_1	m_0	m_2		m_3			m_4

F. Local search strategy

Before the local search for each individual, the scheduling $s_i \in \mathcal{S}$ with the maximum value of $times(s_i)$ is selected as local search element. If there are two or more individuals with the maximum conflict times, the element to be searched is select stochastically among them. Then the offset $s_i.\phi$ of selected s_i is reallocated in its possible offset $s_i.\Phi$. Each reallocated individual is a candidate to update the previous individual. After marking all of the candidate, the individual with minimum score replaces the previous individual.

TABLE VIII shows an example of local search for $indi0$ in Fig II. According to the $T(\mathcal{S})$ of individual 0 in TABLE V, we random select s_0 as the element. Since the possible offset $s_i.\Phi$ of $s_0.\phi$ is 0 and 1 shown in TABLE I, we locate the the scheduling s_0 for message m_0 on its possible offset, gene 0 and gene 1 respectively. After marking two of individual with different offset $s_0.\phi$, we select the individual which score is 2 as the updated $indi0$ because this individual own minimum score among the candidates which just reallocate the offset of s_0.

In our implement, we naively traverse all the possible offset. However for improving efficiency, many other algorithm in local search can be applied, i.e. tabu search, simulated annealing. And different local search strategy may increase the performance of the memetic algorithm.

VII. EVALUATION

A. Experiments Configuration

The network architecture we employed is 2D mesh network which scale is 3×3, 5×5 and 7×7, with 9, 25 and 49 switching nodes respectively. The program of our algorithm is implement in JAVA and is running on a Windows with 4GHz CPU and 12GB memory. We compare the memetic algorithm with general genetic algorithm without local search.

The number of scheduling of messages in our evaluation is from 5 to 50. Each scheduling of message owns its period and delay which is random generated. We test 15 cases in each scale of architecture with different number of messages. The size of population we configure is 100. Therefore there is 100 initial individual. And the maximum iterate times we set is 100. The memetic algorithm will finish if there is a feasible scheduling, otherwise the algorithm will execute until the iterate 100 times. The case which executes exceeds an hour will be regarded as *overtime*.

B. Failed Scheduling Rate

We define the failed scheduling of messages as the minimum number of messages. The messages which failed to scheduling block the scheduling of other messages. If the set of failed communication is removed from the given communication set on the TTNoC, the remaining messages will be scheduled.

Fig 3 shows the rate of failed messages in the set of communication for variable type of test case. It should be noted that the rate is the average rate of the 15 test cases. As can be seen, though the general algorithm spends quite a few time, the rate of failed scheduled messages is much more than the memetic algorithm, especially when the scale of communication set is relatively large. E.g. for the case of 50 messages in 3×3 TTNoC, the failed communication percentage is 17.3% for genetic algorithm while 8% for memetic algorithm. Therefore the local search in the memetic algorithm is the key step in memetic algorithm to significantly improve the performance of genetic algorithm.

And it is clear that the failed scheduling rate decreases when the scale of TTNoC architecture is larger. The reason of this result is that the link resource increases with the growth of architecture, resulting of decreasing the possibility of contention among the messages transmission on the TTNoC. It also presents that the failed communication rate increases when the number of messages on the TTNoC increasing. It is because that the limited link resources have to transmit more messages when there is more communication on the TTNoC.

To compare the performance of the general genetic algorithm and memetic algorithm, we define MA/GA to represent the failed scheduling decreasing comparing MA and GA. TABLE IX depicts the failed scheduling decrease of MA compared with GA. As we can see, The rate of failed scheduling by MA is about 30% of the rate by GA.

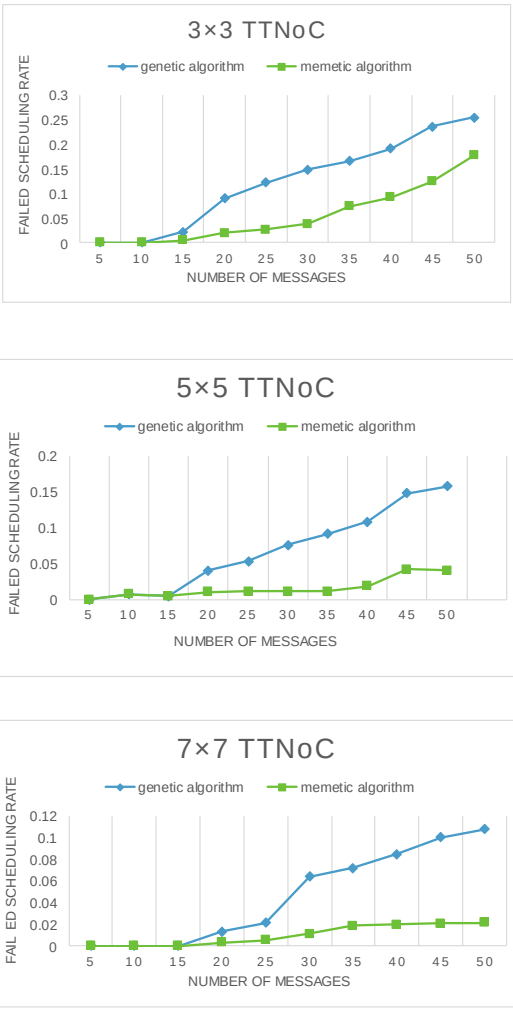

Fig. 3. The rate of failed scheduling

TABLE IX
THE AVERAGE RATE OF FAILED SCHEDULING IN DIFFERENT ARCHITECTURE

Architecture	MA	GA	MA/GA
3×3 TTNoC	0.0557	0.1232	0.4521
5×5 TTNoC	0.0154	0.0685	0.2248
7×7 TTNoC	0.010	0.0465	0.2165
Average	0.0271	0.0794	0.2978

C. Feasible Cases

We consider the number of feasible case in each experiment type. The case is successful if all the communication are scheduled in feasible location without link contention, otherwise we consider the case is infeasible case. Fig 4 presents the result of number of feasible cases after schedule synthesis. For simple test cases with less than 15 communication, almost cases is feasible. But for complexed test cases more than 30 messages, it is hard for the algorithms to find a feasible scheduling.

D. Discussions

Synthesizing a feasible scheduling depends on the number of communication and the scale of TTNoC. It is hard for

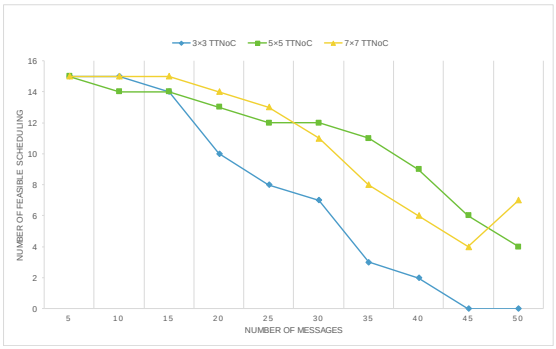

Fig. 4. The number of feasible cases in different test cases

memetic algorithm to synthesize a feasible scheduling when the set of communication is large and/or the TTNoC architecture is small. However, except two reason above, the mapping between a message and a communication node and routing strategy of communications also affect generating a feasible solution. Therefore the designer can manually change the map allocation or the routing strategy to the failed scheduling, and even design a larger network architecture if needed. After these change, the feasible scheduling can be generated by the memetic algorithm iteratively.

VIII. CONCLUSION

This paper introduces a memetic algorithm to resolve the message scheduling problem on the TTNoC. We verify our memetic algorithm with on various random generated messages on different scales of TTNoC. The experiment results shows that our memetic algorithm is efficient to synthesize a scheduling with high percentage of scheduled messages.

In our future work, the mapping between messages and nodes and routing strategy may be integrated to the TTNoC scheduling. And the different TTNoC architecture, i.e. torus, hypercube, and different local search strategy can be also considered in our experiment.

REFERENCES

[1] W. Steiner, G. Bauer, B. Hall, M. Paulitsch, and S. Varadarajan, "Ttethernet dataflow concept," in *Proceedings of The Eighth IEEE International Symposium on Networking Computing and Applications, NCA 2009, July 9-11, 2009, Cambridge, Massachusetts, USA*, 2009, pp. 319–322. [Online]. Available: http://dx.doi.org/10.1109/NCA.2009.28

[2] S. Beji, S. Hamadou, A. Gherbi, and J. Mullins, "Smt-based cost optimization approach for the integration of avionic functions in IMA and ttethernet architectures," in *18th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications, DS-RT 2014, Toulouse, France, October 1-3, 2014*, 2014, pp. 165–174. [Online]. Available: http://dx.doi.org/10.1109/DS-RT.2014.28

[3] M. Hu, J. Luo, Y. Wang, and B. Veeravalli, "Scheduling periodic task graphs for safety-critical time-triggered avionic systems," *IEEE Trans. Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 2294–2304, 2015. [Online]. Available: http://dx.doi.org/10.1109/TAES.2015.140063

[4] H. Kopetz, "GENESYS - A cross-domain architecture for dependable embedded systems," in *5th Latin-American Symposium on Dependable Computing, LADC 2011, São José dos Campos, Brazil, 25-29 April 2011*, 2011, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/LADC.2011.9

[5] J. Huang, J. O. Blech, A. Raabe, C. Buckl, and A. Knoll, "Static scheduling of a time-triggered network-on-chip based on SMT solving," in *2012 Design, Automation & Test in Europe Conference & Exhibition, DATE 2012, Dresden, Germany, March 12-16, 2012*, 2012, pp. 509–514. [Online]. Available: http://dx.doi.org/10.1109/DATE.2012.6176522

[6] W. Steiner, "An evaluation of smt-based schedule synthesis for time-triggered multi-hop networks," in *Proceedings of the 31st IEEE Real-Time Systems Symposium, RTSS 2010, San Diego, California, USA, November 30 - December 3, 2010*, 2010, pp. 375–384. [Online]. Available: http://dx.doi.org/10.1109/RTSS.2010.25

[7] X. Chen, Y. Ong, M. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Trans. Evolutionary Computation*, vol. 15, no. 5, pp. 591–607, 2011. [Online]. Available: http://dx.doi.org/10.1109/TEVC.2011.2132725

[8] S. S. Craciunas and R. S. Oliver, "Combined task- and network-level scheduling for distributed time-triggered systems," *Real-Time Systems*, vol. 52, no. 2, pp. 161–200, 2016. [Online]. Available: http://dx.doi.org/10.1007/s11241-015-9244-x

[9] ——, "Smt-based task- and network-level static schedule generation for time-triggered networked systems," in *22nd International Conference on Real-Time Networks and Systems, RTNS '14, Versaille, France, October 8-10, 2014*, 2014, p. 45. [Online]. Available: http://doi.acm.org/10.1145/2659787.2659812

[10] F. Pozo, W. Steiner, G. Rodríguez-Navas, and H. Hansson, "A decomposition approach for smt-based schedule synthesis for time-triggered networks," in *20th IEEE Conference on Emerging Technologies & Factory Automation, ETFA 2015, Luxembourg, September 8-11, 2015*, 2015, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1109/ETFA.2015.7301436

[11] J. W. Ro, P. S. Roop, and A. Malik, "Schedule synthesis for time-triggered multi-hop wireless networks with retransmissions," in *IEEE 18th International Symposium on Real-Time Distributed Computing, ISORC 2015, Auckland, New Zealand, 13-17 April, 2015*, 2015, pp. 94–101. [Online]. Available: http://dx.doi.org/10.1109/ISORC.2015.24

[12] L. Zhang, D. Goswami, R. Schneider, and S. Chakraborty, "Task- and network-level schedule co-synthesis of ethernet-based time-triggered systems," in *19th Asia and South Pacific Design Automation Conference, ASP-DAC 2014, Singapore, January 20-23, 2014*, 2014, pp. 119–124. [Online]. Available: http://dx.doi.org/10.1109/ASPDAC.2014.6742876

[13] M. Hu, J. Luo, Y. Wang, M. Lukasiewycz, and Z. Zeng, "Holistic scheduling of real-time applications in time-triggered in-vehicle networks," *IEEE Trans. Industrial Informatics*, vol. 10, no. 3, pp. 1817–1828, 2014. [Online]. Available: http://dx.doi.org/10.1109/TII.2014.2327389

[14] C. Scholer, R. Krenz-Baath, A. Murshed, and R. Obermaisser, "Optimal sat-based scheduler for time-triggered networks-on-a-chip," in *10th IEEE International Symposium on Industrial Embedded Systems, SIES 2015, Siegen, Germany, June 8-10, 2015*, 2015, pp. 156–161. [Online]. Available: http://dx.doi.org/10.1109/SIES.2015.7185054

[15] A. Murshed, R. Obermaisser, H. Ahmadian, and A. F. Khalifeh, "Scheduling and allocation of time-triggered and event-triggered services for multi-core processors with networks-on-a-chip," in *13th IEEE International Conference on Industrial Informatics, INDIN 2015, Cambridge, United Kingdom, July 22-24, 2015*, 2015, pp. 1424–1431. [Online]. Available: http://dx.doi.org/10.1109/INDIN.2015.7281942

[16] M. Freier and J. Chen, "Time-triggered communication scheduling analysis for real-time multicore systems," in *10th IEEE International Symposium on Industrial Embedded Systems, SIES 2015, Siegen, Germany, June 8-10, 2015*, 2015, pp. 108–116. [Online]. Available: http://dx.doi.org/10.1109/SIES.2015.7185046

[17] C. Paukovits and H. Kopetz, "Concepts of switching in the time-triggered network-on-chip," in *The Fourteenth IEEE Internationl Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA 2008, Kaohisung, Taiwan, 25-27 August 2008, Proceedings*, 2008, pp. 120–129. [Online]. Available: http://dx.doi.org/10.1109/RTCSA.2008.18

[18] J. Duato, S. Yalamanchili, and L. M. Ni, *Interconnection networks - an engineering approach*. IEEE, 1997.

[19] A. Shpiner, E. Kantor, P. Li, I. Cidon, and I. Keslassy, "On the capacity of bufferless networks-on-chip," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 2, pp. 492–506, 2015. [Online]. Available: http://dx.doi.org/10.1109/TPDS.2014.2310226

[20] P. Mesidis and L. S. Indrusiak, "Genetic mapping of hard real-time applications onto noc-based mpsocs - A first approach," in *Proceedings of the 6th International Workshop on Reconfigurable Communication-centric Systems-on-Chip, ReCoSoC 2011, Montpellier, France, 20-22 June, 2011*, 2011, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/ReCoSoC.2011.5981532

[21] Y. Ong, M. Lim, and X. Chen, "Memetic computation - past, present & future [research frontier]," *IEEE Comp. Int. Mag.*, vol. 5, no. 2, pp. 24–31, 2010. [Online]. Available: http://dx.doi.org/10.1109/MCI.2010.936309