Dynamic Programming-Based Lifetime Reliability Optimization in Networks-on-Chip

Liang Wang, Xiaohang Wang, Terrence Mak

Dynamic Programming-Based Lifetime Reliability Optimization in Networks-on-Chip

Liang Wang[1], Xiaohang Wang[2], and Terrence Mak[1,2]

[1] Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
{lwang, stmak}@cse.cuhk.edu.hk
[2] Guangzhou Institute of Advanced Technology,
Chinese Academy of Sciences, China
xh.wang@giat.ac.cn

Abstract. Technology scaling leads to the reliability issue as a primary concern in Networks-on-Chip (NoC) design. Due to routing algorithms, some routers may age much faster than others, which become a bottleneck for system lifetime. In this chapter, lifetime is modeled as a resource consumed over time. A metric lifetime budget is associated with each router, indicating the maximum allowed workload for current period. Since the heterogeneity in router lifetime reliability has strong correlation with the routing algorithm, we define a problem to optimize the lifetime by routing packets along the path with maximum lifetime budgets. A dynamic programming-based lifetime-aware routing algorithm is proposed to optimize the lifetime distribution of routers. The dynamic programming network approach is employed to solve this problem with linear complexity. The experimental results show that the lifetime-aware routing has around 20%, 45%, 55% minimal MTTF improvement than XY routing, NoP routing, and Oddeven routing, respectively.

Keywords: Reliability, Networks-on-Chip, Routing algorithm, Dynamic programming

1 Introduction

Networks-on-Chip (NoC) is emerging as an efficient communication infrastructure for connecting resources in many core system. NoC is composed of routers interconnected through a network. NoC provides communication fabrics for data transmission among cores. The data transmission is in the form of packets, which is divided into flits and routed by routers. In NoC, routing algorithm provides a protocol for routing the packets. In other words, the pathways of the packets are determines by a routing algorithm. Generally, routing algorithms are classified into deterministic routing and adaptive routing. Deterministic routing algorithm provides a fixed path given source and destination. Compared to deterministic routing algorithm, adaptive routing algorithm is more flexible. The pathway of a packet can dynamically adapt to NoC traffic or other conditions. In this chapter,

we exploit an adaptive routing algorithm to optimize the lifetime reliability of NoC.

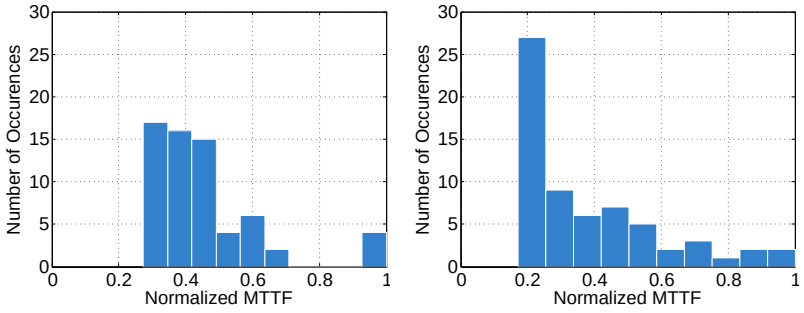
With shrinking feature size and increasing transistor density, reliability issue is becoming a primary concern for chip design. The failure rate of electronic components increases 316% as the features size decreases 64% [27]. Along with shrinking feature size, power density of chips increases exponentially, leading to overheat. High temperature also greatly reduces the lifetime of a chip. Dynamic thermal management (DTM) techniques such as dynamic voltage and frequency scaling (DVFS) [13], adaptive routing [2] are employed to address the temperature issues. The temperature is maintained below a limit to ensure the reliability of a chip. Dynamic reliability management (DRM) is first proposed in [26], aiming at ensuring a target lifetime reliability at better performance.

The reliability of NoC depends on the routers. The lifetime reliability of a router has strong correlation with the routing algorithm because the lifetime reliability is relevant to operating conditions and temperature, which are affected by the routing algorithm. We conduct a case study to show the distribution of routers reliability under two different routing algorithms, XY and Oddeven. The case study is evaluated in 8×8 2D mesh NoC. The detailed description of simulation setup is referred to Section 5. The lifetime, measured in MTTF metric (mean time to failure), is normalized to the maximum one. The results are presented in Figure 1, which shows the number of occurrences in different MTTF ranges. For both routing algorithms, there is a heterogeneity observed among the routers. Especially for Oddeven routing, the minimum MTTF of router is even less than 20% of the maximum one. It suggests that the minimum MTTF router is aging more than 5 times faster than the maximum MTTF router. The unbalanced lifetime distribution would become a bottleneck for the lifetime of system. Furthermore, the two distribution functions differ in slop for XY and Oddeven, indicating the correlation of router reliability and routing algorithms.

The above example indicates routing paths can be a control knob to optimize the router reliability. In this chapter, we apply dynamic reliability management to NoC and propose a lifetime-aware routing to optimize the lifetime reliability of NoC routers. Lifetime is modeled as a resource consumed over time. A lifetime budget is defined for each router, indicating the maximum allowed workload for current time. We define a longest path problem to optimize the router lifetime by routing packets along the path with maximum lifetime budgets. The problem is solved by dynamic programming approach with linear time complexity. The key idea is to use lifetime budget as the cost for dynamic programming. Moreover, a low cost hardware unit is implemented to accelerate the lifetime budget computation at runtime.

This chapter is an extension of previous work [28] by adding more detail descriptions, substantial analysis and modified experiments. The main contributions of this chapter include:

(1) Define a lifetime budget for each router, indicating the maximum allowed workload for current period.



(a) XY routing (b) Oddeven routing

Fig. 1. A case study for motivation. In 8×8 NoC, the Normalized MTTF of routers is evaluated under different routing algorithms

(2) Define a problem to optimize the lifetime by routing packets along the path with maximum lifetime budgets.

(3) Propose a lifetime-aware routing algorithm, which solves the problem through a dynamic programming approach with linear time complexity.

The remainder of the chapter is organized as follows. Section 2 briefly introduces the related work. Section 3 discusses the DRM and defines the lifetime budget for a router. Section 4 presents the adaptive routing, including problem formulation and routing algorithm. Section 5 analyzes the experimental results and Section 6 concludes this chapter.

2 Related Work

There are two kinds of failures in ICs: extrinsic failures and intrinsic failures. Extrinsic failures are caused by manufacturing defects and occur with a decreasing rate over time. Intrinsic failures are related with wear-out and are caused due to operation conditions within the specified conditions, e.g. temperature, current density, *etc.* In this chapter, we focus on long-term reliability management of routers, and only consider intrinsic failures. The failure mechanisms for intrinsic failures include electro migration (EM), time-dependent dielectric breakdown (TDDB), stress migration (SM), Negative Bias temperature instability (NBTI) and thermal cycling (TC). A reliability model named RAMP is proposed in [26], which combines various failure mechanism models using Sum-of-failure method.

In the failure mechanism models, lifetime reliability is highly related to temperature. Most prior studies consider thermal issues, with the objectives to balance the temperature or to take temperature as a constraint [23][13][2]. Mulas *et al.* [23] employed a task migration approach to redistribute power dissipation such that the temperature of multiprocessor system is balanced. Hanumaiah *et al.* [13] adopted DVFS to maintain the temperature of multiprocessor system under a constraint. Al-Dujaily *et al.* [2] proposed to balance the temperature of

NoC by a thermal-aware routing algorithm. However, the thermal techniques neglect other factors on reliability, such as switch activity, operating frequency, *etc.* The lifetime could not be effectively balanced.

Dynamic reliability management (DRM), proposed in [26], [19], regards the lifetime as a source that could be consumed. Reliability management is mainly studied for single-core processor or multi-core processors through various solutions, such as task mapping [14], frequency control [25], reliability monitoring and adaptation [22], *etc.* Hartman [14] proposed to dynamically manage the lifetime of chip multiprocessors through run-time task mapping. The task mapping obtains data from on-chip reliability sensors and adapts to changing lifetime distribution in the system at run-time. Shi *et al.* [25] explored DRM for both single-core and multi-core processors. The overall performance expressed as frequency policies is maximized under soft thermal constraint. Mercati *et al.* [22] proposed a DRM policy based on a two level controller. The controller monitors system reliability on a long time scale and adapts operating conditions on a short time scale. The multi-core system adapts operating conditions with DVFS such that a predefined target lifetime is satisfied.

Since NoC is becoming more important for multi-core system interconnection, reliability management in NoC domain is attracting increasing attentions. Some studies make attempt to improve the NoC reliability through microarchitecture design. A wear-resistant router microarchitecture is designed in [17] to improve reliability of routers. However, they did not consider the routing algorithm impacts on the router lifetime. Task mapping is another solution to improve NoC reliability. A compile-time task mapping algorithm is proposed in [12] to balance the MTTF of NoC. However, at runtime the tasks are mapped on NoC-based MPSoC without considering the variation of runtime operating conditions. The reliability of NoC can also be improved through routing algorithms. Bhardwaj *et al.* proposed an aging-aware adaptive routing algorithm for NoC [7][6]. They introduced an aging model that defines stressed links and routers, in which the traffic of a router or link exceeds the upper limit called Traffic Threshold per Epoch (TTpE). However, the routing algorithm actually reduces the workloads of routers with high utilization, which may not exhibit the most aging effects. Different from their works, we directly apply reliability management to NoC, and propose a lifetime-aware routing algorithm to balance the lifetime distribution of NoC routers at runtime. The routing algorithm is based on the dynamic programming (DP) approach, which is proposed by Mak *et al.* [21]. The dynamic programming approach is proposed for adaptive routing, in which the shortest path problem is solved optimally. The dynamic programming based adaptive routing has already been applied in congestion avoidance [21], fault tolerance [35], thermal management [2], *etc.*

3 Lifetime Budget Definition

For lifetime-aware routing algorithm, the lifetime reliability of routers should be provided for the algorithm to update routing decisions. There are mainly two methods to estimate lifetime reliability:

(1) Reliability is estimated through operating conditions history [34]. Using existing mathematical failure models, aging is periodically computed. At runtime, the operating conditions are monitored and provided for lifetime estimation.

(2) Aging sensors are used to monitor the aging effects of transistors [18]. For example, NBTI sensors are exploited to monitor the variation of threshold voltage, as the NBTI causes an increase on the threshold voltage of PMOS transistors. However, besides NBTI, the wear-outs of transistors are also incurred by other failure mechanisms such as EM, which could not be monitored by sensors explicitly.

In fact, both methods can be used for our lifetime-aware routing algorithm because the lifetime-aware routing is independent of lifetime estimation. The primary objective of this chapter is on lifetime-aware routing for lifetime optimization. We adopt the first method for lifetime estimation, i.e, the lifetime of routers are estimated from temperature and workload stresses history. We also present a hardware implementation for lifetime estimation in Section 4.5.

For long term reliability management of routers, we only consider wear-out related faults. The failure rate, a metric for lifetime reliability, keeps almost constant if the operating conditions (e.g. constant current, temperature, frequency and voltage) keep unchanged. The mean time to failure (MTTF) is inverse of failure rate when the operating conditions are constant. The MTTF due to EM is based on Black's equation [1] as follows

$$MTTF \propto (J - J_{crit})^{-n} \exp\left(E_a/kT\right) \tag{1}$$

where J is the current density; J_{cris} is the critical current density; E_a is the activation energy; k is the Boltzmann's constant; T is temperature. n is a constant depend on interconnect metal used. However, this equation only assumes steady operating conditions, which is not realistic. The operating conditions (temperature, current density) are usually varying due to workload variation. Lu. *et al.* [19] derived MTTF under time-varying current density and temperature stresses as

$$T^f = \frac{A}{E\left[j(t)\left(\frac{exp(\frac{-Q}{kT(t)})}{kT(t)}\right)\right]} \tag{2}$$

where A is a constant related to the structure. $j(t)$ is current, and $T(t)$ is temperature. The varying failure rate, also called lifetime consumption rate, is denoted as $\lambda(t) = j(t)\left(\frac{exp(\frac{-Q}{kT(t)})}{kT(t)}\right)$. The MTTF is the inverse of failure rate expectation value. Eq. 2 is based on the assumption of Electromigration (EM) failure mechanism. We only consider EM because EM is the primary aging factor for

interconnection, and the reliability of both routers and links are closely related to interconnection.

Based on Eq. 2, we also derive an equation as an approximation relationship between $\lambda(t)$ and the routers workload. The current is $j(t) = \frac{CV_{dd}}{WH} \times f \times p$ [26], where p is switch activity. The voltage and frequency are assumed constant. In a router, switch activity is proportional to the incoming rate of a router because the incoming flits are assumed the only stimuli to the allocator. The failure rate can be represented as follows:

$$\lambda(t) \propto d(t) \left(\frac{exp(\frac{-Q}{kT(t)})}{kT(t)} \right) \tag{3}$$

where $d(t)$ is the flits incoming rate at time t. The flits incoming rate is the number of flits passing through the router per unit time. The incoming rate also stands for the workload of a router. It is assumed that the ports of the incoming flits are not considered. The equation provides an approximated relationship between the lifetime and routers workloads.

Another equation derived in [19] is $\int_0^{T^f} r(t)dt = C$, where C is a constant. In the equation, lifetime is modeled as a resource consumed over time. As suggested by [19], we define a lifetime budget for each router, denoted as

$$LB(t) = \int_0^t (\lambda_{nominal} - \lambda(t))dt \tag{4}$$

where $\lambda_{nominal}$ is derived from the specified expected lifetime, indicating the constant lifetime consumption rate under nominal conditions. $\lambda_{nominal}$ is the inverse of expected MTTF, i.e., $\lambda_{nominal} \cdot T^f = C$. If $LB(t) > 0$, the expected lifetime could satisfy the predefined constraint, and vice versa. The failure rate is related to operating conditions, i.e., temperature and workload, which are monitored periodically. Under discrete monitored conditions, the lifetime budget is represented as

$$LB(n) = \begin{cases} 0, & \text{if } n \text{ is } 0 \\ LB(n-1) + \lambda_{nominal} - \lambda(n), & \text{Otherwise} \end{cases} \tag{5}$$

where $LB(n)$ and $\lambda(n)$ are the lifetime budget and failure rate respectively at the n-th time interval. The lifetime budget indicates the maximum allowed failure rate for current the period. Therefore, the lifetime is modeled as a source of routers to be consumed over time. The router with higher workloads consumes the lifetime source at a faster speed.

From the perspective of packets, the selected path determines the workloads of the routers along the path. Therefore the routing algorithm, which determines the routing paths, plays an important role in the lifetime distribution of routers. In following sections, we propose a lifetime-aware routing algorithm to balance the lifetime distribution of routers.

4 Lifetime-Aware Adaptive Routing

Since MTTF or failure rate of a router is relevant to the flits incoming rate and temperature. We propose to balance the MTTF of routers through an adaptive routing algorithm. In this section, we first define a problem for lifetime reliability optimization and present the dynamic programming formulation for the problem. Then we propose an adaptive routing algorithm for lifetime reliability optimization. Table 1 summarizes all notations in this section.

Table 1. Notations

Symbols	Semantics
$\mathcal{V}$	A set of nodes in network $\mathcal{G}$
$\mathcal{A}$	A set of edges in network $\mathcal{G}$
LB_i	Lifetime budget of the router i
$C_{u,d}$	Cost of edge $u \to d$
$V(s,d)$	The optimal cost from s to d in LP form
$V^*(s,d)$	The optimal cost from s to d in DP form
$V^{(k)}(s,d)$	The expected cost for s after k steps
$P_{s,d}$	Available paths from s to d
r_i	The i-th router along a path from s to d
$\mu(d)$	The optimal routing direction to node d
$N(j)$	The neighbor node in the direction j

4.1 Problem Definition

To balance the lifetime distribution, the lifetime-aware adaptive routing aims to find a path with maximum lifetime budget from designated path sets for each packet. Therefore we formulate a longest path problem as follows. Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with $n = |\mathcal{V}|$ nodes, $m = |\mathcal{A}|$ edges, and a cost associated with each edge $u \to v \in \mathcal{A}$, which is donated as $C_{u,v}$. Given two nodes $s, d \in \mathcal{V}$, $P_{s,d}$ is the set of minimal distance paths from s to d. The cost of a path $p = \langle s = v_0, ..., d = v_k \rangle \in P_{s,d}$, from s to d, is the sum of the costs of its constituent edges: $Cost(p) = \sum_{i=0}^{k-1} C_{i,i+1}$. We aim to find the path with the maximum cost of the path, denoted as $V(s,d)$. The problem can be formally formulated as a linear optimization problem. Let u be a neighbor node of s, and on the one of the minimal distance paths. We have a constraint

$V(s,d) \geq V(s,u) + C_{u,d}$. We can obtain the following linear programming:

$$\begin{aligned} \text{maximize} \quad & \sum_{\forall s \in \mathcal{V}} V(s,d) \\ \text{subject to} \quad & V(s,d) \geq V(u,d) + C_{s,u} \\ & V(d,d) = 0 \end{aligned} \tag{6}$$

The above formation yields the optimal path from any nodes s to the destination node d, known as multiple-source single-destination longest path problem. With the nodes corresponding to the routers, the key idea of the adaptive routing is to use lifetime budget as the cost for the path, denoted as

$$C_{r_i, r_{i+1}} = LB_i \tag{7}$$

LB_i is the lifetime budget of the i-th router r_i. The total lifetime budgets along the path $p = \langle r_0 = s, ..., r_{k-1} = d \rangle$ is

$$C_{s,d} = \sum_{i=0}^{k-1} LB_i \tag{8}$$

Taking lifetime budget as the cost, the problem is to find a path with maximum lifetime budgets.

4.2 Dynamic Programming-based Formulation

Background of Dynamic Programming Dynamic programming (DP) is an optimization technique which was first introduced by Richard Bellman in the 1940s [5]. DP has been applied in a variety practical problems, in which the main complex problem can be broken into simpler subproblems. It provides a systematic procedure for determining the optimal combination of decisions which takes much less time than naive methods. In contrast to other optimization techniques, such as linear programming (LP), DP does not provide a standard mathematical formulation of the algorithm. Rather, DP is a general type of approach to problem solving, and it restates an optimization problem in recursive form, which is known as Bellman equation. The optimization or decision-making problems can be expressed in a recursive form as follows:

$$V_i(t) = \max_{\forall k} \{R_{i,k}(t) + V_k(t)\}, \ \forall i \tag{9}$$

where $V_i(t)$ is the expected reward of the i-th state and $R_{i,k}(t)$ is the reward of transition from state i to state k.

DP Formulation The problem to find a path with maximum cost can also be stated in the form of dynamic programming, which defines a recursive process in step k. To obtain the optimal cost from node s to d, the process requires the

notion of cost-to-go function, which is expect cost from node s to d. Based on the Bellman equation, the expected cost is updated recursively until the optimality criteria is reached. The Bellman equation can be expressed as

$$V^{(k)}(s,d) = \max_{\forall u \in V} \left\{ V^{(k-1)}(u,d) + C_{s,u} \right\} \tag{10}$$

$V^{(k)}(s,d)$ is the cost from s to d at the k-th iteration. The cost $C_{s,u}$ is associated with lifetime budget. Initially, $u = d$ and $V^{(0)}(d,d) = 0$. Then Eq. 10 is solved recursively and the recursion is expanded from s to d. After k iterations, the optimal cost from s to d is $V^*(s,d)$, which is maximum among all minimal distance paths $P_{s,d}$. The optimal cost is represented as following equation:

$$V^*(s,d) = \max_{\{r_0=s,\ldots,r_{k-1}=d\} \in P_{s,d}} \left\{ \sum_{i=0}^{k-1} LB_i \right\} \tag{11}$$

At each node s, the optimal decision that leads to the optimal path to d can be obtained from the argument of the maximum operator at the Bellman equation as follows:

$$\mu(d) = arg \max_{\forall j} \{ V^*(N(j),d) + LB_s \} \tag{12}$$

where j is the optimal decision, which represents the optimal output port or output direction. $N(j)$ is the neighbor node in the direction j. LB_s is the lifetime budget of node s.

Compared with linear programming, the dynamic programming presents an opportunity for solving the problem using parallel architecture and can greatly improve the computation speed.

4.3 Lifetime-Aware Adaptive Routing Algorithm

We propose a dynamic programming-based lifetime-aware adaptive routing algorithm, which is outlined in Algorithm 1. This algorithm outputs the direction to be taken for current node s. First, according to the positions of local node and destination node, the available directions D_s are restricted to the minimal distance paths to destination (line 1). If the local node is the destination, the optimal cost is 0 and the routing direction is local port. Given an available direction $j \in D_s$, the expected cost is computed by adding up the local cost LB_s and the optimal cost from neighbor node $N(j)$ to d (lines 6-8). The maximum cost is obtained by taking the maximum value from all $V_j(s,d)$, which are the costs of the paths that local node s takes direction j (line 9). Finally, the optimal direction $\mu(d)$ is obtained from the argument of the maximum operator (line 10). The dynamic programming-based adaptive algorithm outputs an optimal direction for each router. In the algorithm, the loop is realized in dynamic programming network. The optimal value for local node is propagated to the all neighbor nodes through dynamic programming network. The computational-delay complexity can be reduced to linear.

Algorithm 1 Lifetime-Aware Adaptive Routing

Definitions s: local node;
D_s: set of directions to minimal distance paths;
$N(j)$: the neighbor node in the direction j;
$V_j(s,d)$: the cost of the path taking direction j for s;
Input d: destination node;
$V^*(N(j),d)$: the optimal cost from $N(j)$ to d;
LB_s: lifetime budget of node s,
Output $\mu(d)$: the optimal routing direction to d;
$V^*(s,d)$: the optimal cost from s to d.

Calculate available direction D_s according to positions of s and d
if $s = d$ **then**
  $V^*(s,d) = 0$
  $\mu(d) = LOCAL$
else
  for all directions $j \in D_s$ **do**
    $V_j(s,d) = V^*(N(j),d) + LB_s$
  end for
  $V^*(s,d) = \max_{\forall j} V_j(s,d)$
  $\mu(d) = arg\max_{\forall j} V_j(s,d)$ ▷ Update optimal routing directions
end if

In this chapter, the routers are assumed wormhole flow control without virtual channel. Deadlock can effectively be avoided by adopting one of the deadlock-free turn model. We adopt west-first turn model for deadlock avoidance [11].

4.4 Dynamic Programming Network

The dynamic programming network, introduced by Mak *et al.* [21], is composed of distributed computation units and links. Figure 2 presents an example of 3×3 dynamic programming network. The dynamic programming network is coupled with NoC. Each computation unit implements the DP unit equations e.g. longest path calculations, and propagates the numerical solution to neighbor units. In addition, routing tables are implemented in routers. Algorithm 1 presents the operations required for updating the routing directions using the DP unit. The routing table will be updated periodically by the DP unit. For each router, the temperature and flits incoming rate are also monitored periodically. Failure rate is computed through the lifetime budget computation unit, which is presented in Section 4.5. According to the computed failure rate and nominal failure rate, the lifetime budget is updated. The lifetime budget values also propagated to the DP units as the DP costs. The dynamic programming network quickly resolves the optimal solution and passes the control decisions to routers, then the routing tables are updated.

The DP network presents several features to NoC:

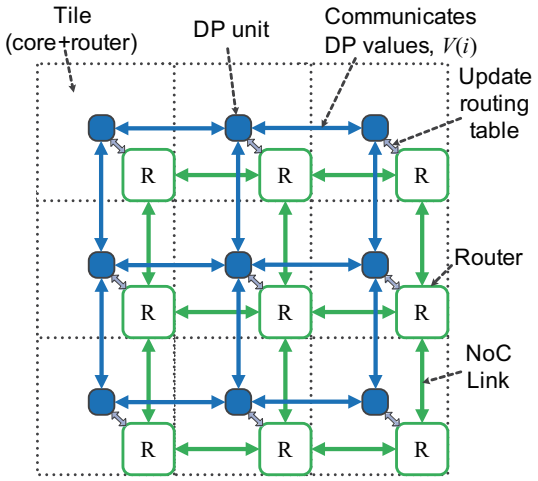


Fig. 2. An example of 3×3 dynamic programming network coupled with NoC

(1) The distributed units enable a scalable monitoring functionality for NoC. Each unit monitors local information and communicates with neighbor units, achieving a global optimization.

(2) The DP network can provide a real-time response without consuming data-flow network bandwidth due to the simplicity of the the computational unit.

(3) The DP network provides an effective solution to the optimal routing.

To converge to the optimal solution, the delay of DP network depends on the network topology. Mak *et al.* have concluded that the network convergence time is proportional to the network diameter, which is the longest path in the network [20].

4.5 Lifetime Budget Runtime Computation

The failure rate computation is an exponential function, not applicable for runtime computation. Similar to the methods proposed in [33], we use lookup tables that fit with Eq. 3 to pre-calculate failure rate. The runtime computation process is accelerated. To compute the lifetime budget at runtime, we design a hardware unit called lifetime budget computation unit (LBCU). The architecture of LBCU is presented in Figure 3. The temperature related part $\left(\frac{exp(\frac{-Q}{kT(t)})}{kT(t)}\right)$ is pre-computed and kept in a lookup table. Each entry is corresponding to a temperature range. Another potential problem is that it may require much area to multiply with the incoming flit rate. Instead of computing the multiplication at the end of each time interval, we compute the failure rate per cycle. As shown in Figure 3, E, S, W, N, L are from 5 ports of a router, indicating whether there is a flit in current cycle. The failure rate per cycle is computed by multiplying the

lookup table result with the sum of the ports. Because the maximum number to be multiplied is 5, the multiplication is only achieved by shifting and addition in stead of a multiplier. A counter is used to judge if it reaches the end of the time interval. At the end of the time interval, the lifetime budget is attained through addition. Despite of a little accuracy loss, the lifetime budget computation is accelerated at runtime while only some basic logic units are used. The implementation of LBCU will be evaluated in terms of area in Section 5.7.

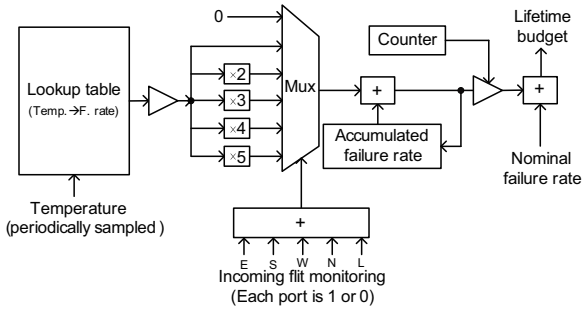


Fig. 3. Lifetime Budget Computation Unit

5 Experimental Results

5.1 Experimental Setup

Experiments are performed using Noxim simulator, which is an open source SystemC simulator for mesh-based NoC. In Noxim, the power of routers are modeled using ORION 2.0 NoC power simulator [16]. To model the temperature, we adopt HotSpot thermal model [15]. The thermal configuration is the default configuration of HotSpot. To be more accurate, we adopt the floorplan of Tilera64 processor [4] as the input of HotSpot. The frequency of NoC is configured 1 GHz. The simulation runs for 10^7 cycles. The time interval for temperature monitoring is 5000 clock cycles. And the routing tables and lifetime budgets are only updated at the end of each time interval. We employ Electromigration as the failure mechanism. The buffer depth is 10 flits and the packet size is 5 flits.

Table 2. Benchmarks

PARSEC	streamcluster, swaptions, ferret, fluidanimate, blackscholes, freqmine, dedup, canneal, vips
SPLASH-2	barnes, raytrace

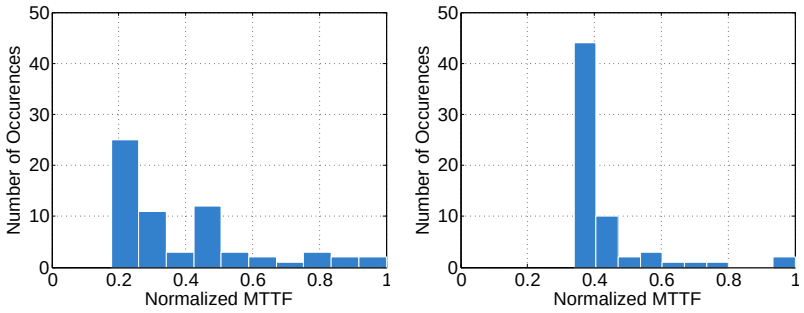
Table 3. Simulator setup

Number of cores	64 (MIPS ISA 32 compatible)
L1 D cache	16KB, 2-way, 32B line, 2 cycles, 2 ports, dual tags
L1 I cache	32KB, 2-way, 64B line, 2 cycles
L2 cache (shared) MESI protocol	64KB slice/node, 64B, 6 cycles, 2 ports
Main memory	2GB
Data packet size	5 flits
Meta packet size	1 flit
NoC flit size	72-bit
NoC VC number	4
NoC buffer	5 × 10 flits

In the experiments, we compare the lifetime-aware routing algorithm with XY routing, NoP routing and Oddeven routing, respectively. The NoP routing algorithm, a congestion-aware routing, is the west-first turn model with neighbors-on-path (NoP) selection scheme; the Oddeven routing is the oddeven turn model [10] with random selection scheme. Besides, the evaluations are also performed over a suite of benchmarks: 9 benchmarks in PARSEC [8] and 2 benchmarks in SPLASH-2 [32]. The benchmarks are listed in Table 2. This experiments are performed in an in-house developed simulator [30]. The configurations for the simulator are listed in Table 3.

5.2 MTTF Distribution

As comparisons with the case study mentioned in Section 1, we evaluate the MTTF distribution of the lifetime-aware routing and NoP routing. The injection



(a) NoP routing (b) Lifetime-aware routing

Fig. 4. MTTF distribution of NoP routing and lifetime-aware routing

Table 4. Minimal MTTF comparisons under different routing algorithms (hours).

NoC size	XY	NoP	Oddeven	Lifetime	MTTF improvement (Lifetime vs.)		
					XY	NoP	Oddeven
8x8	577490	453085	435682	683209	18.3%	50.8%	56.9%
10x10	321487	264756	253358	393711	22.4%	48.7%	55.4%
12x12	173618	144960	133637	203117	16.9%	40.1%	52.0%

rate is also 0.005 flits/cycle; the NoC size is 8×8. Figure 4 presents the histogram of lifetime distribution. For the lifetime-aware routing, the MTTF distribution is more concentrated than others, namely, the MTTF is more evenly distributed.

5.3 Minimal MTTF Evaluation

The minimal MTTF router is the router with the highest probability to wear out. Because the minimal MTTF is critical for the system lifetime, we evaluate the minimal MTTF of routers, expressed in $\mathbf{min}\{MTTF_i\}$. The evaluation is under synthetic traffic. The traffic pattern is set random and the injection rate is set 0.005 flits/cycle. The routing algorithms are also compared in different NoC size, 8×8,10×10, 12×12. The results are shown in Table 4. In the table, the minimal MTTF value is evaluated. The evaluation metric is hour. It can be observed that the lifetime-aware routing has around 20%, 45%, 55% minimal MTTF improvement than XY routing, NoP routing, Oddeven routing, respectively. Additionally, the minimal MTTF also decreases dramatically with NoC size, because the workloads of routers increase with the area of NoC. The MTTF improvement against XY routing is relatively smaller as the XY routing also brings relatively less traffic for the routers in the central region.

We also evaluate the minimal MTTF with real benchmarks. The experimental results are demonstrated in Figure 5. Here all the minimal MTTF values are normalized to the values under lifetime-aware routing. The minimal MTTF under lifetime-aware routing is highest, which is consistent with previous results. In addition, the minimal MTTF under different benchmarks varies a lot because the workloads are inherently unbalanced.

5.4 NoC Overall MTTF Evaluation

We take NoC as a whole and evaluate the overall MTTF of NoC. This based on the assumption that NoC fails when a router fails. Therefore, the failure rate of NoC is the sum of all routers, denoted as $\lambda_{NoC} = \sum_{i=1}^{N} \lambda_i$. The MTTF of NoC is calculated according to Eq. 2. We evaluate the MTTF of NoC under real benchmarks with 4 different routing algorithms. The results are presented in Figure 6. From the results, we found that the lifetime-aware routing leads to around 5% NoC MTTF improvement due to its better lifetime distribution.

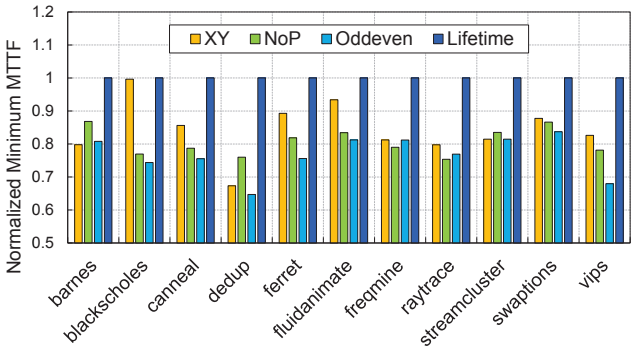


Fig. 5. Minimal MTTF evaluation with real benchmarks

This is because the overall workloads are almost the same for different routing algorithms, while the lifetime-aware routing algorithm also leads to better temperature distribution.

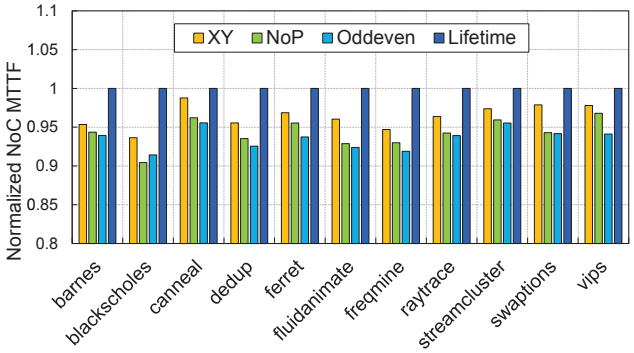


Fig. 6. NoC MTTF evaluation with real benchmarks

However, the overall MTTF cannot effectively reflect the reliability of routers. The unbalanced lifetime distribution would make some routers age much faster despite of the small differences of overall MTTF. An example is illustrated in [24], showing that overall MTTF metric is not adequate for overall reliability specification.

5.5 Variance of MTTF

Besides overall MTTF, we also use the MTTF variance metric to show that the lifetime-aware routing distributes the lifetime more evenly. The results are shown

in Figure 7. In the figure, the MTTF variance is normalized for comparisons. The lifetime-aware routing algorithm exhibits the less variance, showing that the lifetime distribution is more balanced.

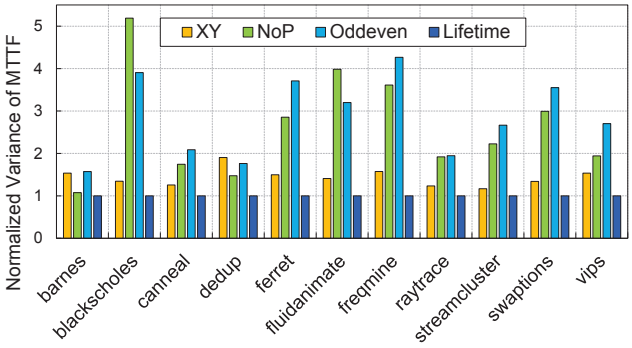


Fig. 7. Variance of MTTF comparison with real benchmarks

5.6 Average Packets Delay Comparison

To evaluate the impacts on the global average delay, the lifetime-aware routing is also compared with the other three routing algorithms. The global average delay is evaluated with random traffic pattern. The buffer size is configured 10 flits. The comparisons are under flits injection rate from 0.01 to 0.17 flits/cycle. The experimental results are shown in Figure 8. The delay is measured in cycles. It can be observed that the saturated flit injection rate of the lifetime-aware routing is around 0.10 flits/cycle, which is 0.02 less than oddeven routing, 0.03 less than NoP routing, and 0.04 less than XY routing. Therefore, when the injection rate is less than 0.10 flits/cycle, these routing algorithms have similar performance in terms of average packet delay. However, the lifetime-aware routing achieves longer life while has smaller saturation point. It is concluded there is a trade-off between lifetime and performance (average packet delay).

5.7 Hardware Evaluation

We implement lifetime budget computation unit (LBCU) with Verilog HDL and compare LBCU with router in terms of area. The lookup table of LBCU contains 64 entris to keep pre-computed values, which corresponds to different temperature ranges. The size of each entry is 32 bits. The registers for lifetime budget and failure rate value are 32 bits. The router is open-source and developed by Becker [3]. The frequency is 1 GHz. The router is 5-ports input-buffered with wormhole flow control. The buffer size is 4 flits; the flit size is 75 bits.

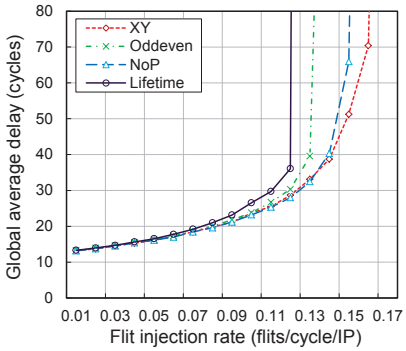


Fig. 8. Global average delay comparison

They are synthesized using Synopsys Design Compiler under 45nm TSMC library. The areas of router and LBCU are 29810 μm^2 and 1529 μm^2 respectively. It can be concluded that LBCU leads to around 5.13% increase in terms of area. In other words, LBCU can be integrated with NoC with low overhead. In addition, the cost of dynamic programming network is not evaluated in this work. The detail evaluation for dynamic programming network can refer to [21].

6 Conclusions and Future work

In this chapter, we propose a dynamic programming-based lifetime-aware routing algorithm for NoC reliability management. First, we define a lifetime budget metric for each router. With this metric, a problem is defined to optimize the lifetime by routing packets along the path with maximum lifetime budgets. We propose a lifetime-aware routing algorithm using dynamic programming approach. Finally, the lifetime-aware routing algorithms are evaluated in synthetic traffic and real benchmarks. The experimental results show that the lifetime-aware routing can distribute the lifetime of routers more evenly. The lifetime-aware routing has around 20%, 45%, 55% minimal MTTF improvement than XY routing, NoP routing, Oddeven routing, respectively.

In the future, we plan to optimize both the lifetime distribution of routers and the average packet latency. This is because we observe that the lifetime-aware routing algorithm lowers the performance in terms of average packet delay. A hybrid routing algorithm will probably be proposed taking consideration of both packet delay and lifetime of routers. Similar to [22], the lifetime is optimized in long-term scale while the performance is optimized in shor-term scale. Thus the lifetime can be improved without having much impact the performance. Another possible future work is to exploit the traffic throttling [9] or DVFS in NoC to maintain the MTTF of NoC above an expected value. This is because the lifetime reliability depends on the voltage, frequency and switching activity. The problem can be defined as maximizing performance given fixed lifetime

budget. This is similar to the power budgeting problem [31] [29] which maximizes performance under limited power budget. However, the lifetime budgeting is different as the aging process is in a long-term scale. Therefore, the strategies for lifetime budgeting is possibly quite different from power budgeting. In the future work, we will exploit novel strategies for lifetime budgeting problem.

Acknowledgment

This research program is supported by the Natural Science Foundation of China No. 61376024 and 61306024, Natural Science Foundation of Guangdong Province No. S2013040014366, and Basic Research Programme of Shenzhen No. JCYJ20140417113430642 and JCYJ20140901003939020.

References

1. Failure mechanisms and models for semiconductor devices. JEDEC Publication (2003)
2. Al-Dujaily, R., Mak, T., Lam, K.P., Xia, F., Yakovlev, A., Poon, C.S.: Dynamic on-chip thermal optimization for three-dimensional networks-on-chip. Comput. J. 56(6), 756–770 (2013)
3. Becker, D.U.: Efficient microarchitecture for network-on-chip routers. In: PhD thesis, Stanford University (2012)
4. Bell, S., Edwards, B., Amann, J., et al.: Tile64 - processor: A 64-core soc with mesh interconnect. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC). pp. 88–598 (2008)
5. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton, NJ, USA (1957)
6. Bhardwaj, K., Chakraborty, K., Roy, S.: An milp-based aging-aware routing algorithm for nocs. In: Proceedings of Design, Automation Test in Europe Conference Exhibition (DATE). pp. 326–331 (2012)
7. Bhardwaj, K., Chakraborty, K., Roy, S.: Towards graceful aging degradation in nocs through an adaptive routing algorithm. In: Proceedings of 2012 49th ACM/EDAC/IEEEDesign Automation Conference (DAC). pp. 382–391 (2012)
8. Bienia, C., Kumar, S., Singh, J.P., Li, K.: The parsec benchmark suite: Characterization and architectural implications. In: Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques (PCAT). pp. 72–81 (2008)
9. Chang, K., Ausavarungnirun, R., Fallin, C., Mutlu, O.: Hat: Heterogeneous adaptive throttling for on-chip networks. In: Proceedings of IEEE 24th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). pp. 9–18 (2012)
10. Chiu, G.M.: The odd-even turn model for adaptive routing. IEEE Trans. Parallel Distrib. Syst. 11(7), 729–738 (2000)
11. Dally, W.J., Seitz, C.L.: Deadlock-free message routing in multiprocessor interconnection networks. IEEE Transactions on Computers (TC) 36(5), 547–553 (1987)
12. Das, A., Kumar, A., Veeravalli, B.: Reliability-driven task mapping for lifetime extension of networks-on-chip based multiprocessor systems. In: Proceedings of Design, Automation Test in Europe Conference Exhibition (DATE), 2013. pp. 689–694 (2013)

13. Hanumaiah, V., Vrudhula, S.: Temperature-aware dvfs for hard real-time applications on multicore processors. IEEE Transactions on Computers 61(10), 1484–1494 (2012)
14. Hartman, A.S., Thomas, D.E.: Lifetime improvement through runtime wear-based task mapping. In: Proceedings of the Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis. pp. 13–22 (2012)
15. Huang, W., Ghosh, S., Velusamy, S., Sankaranarayanan, K., Skadron, K., Stan, M.: Hotspot: a compact thermal modeling methodology for early-stage vlsi design. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 14(5), 501–513 (2006)
16. Kahng, A., Li, B., Peh, L.S., Samadi, K.: Orion 2.0: A power-area simulator for interconnection networks. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20(1), 191–196 (2012)
17. Kim, H., Vitkovskiy, A., Gratz, P.V., Soteriou, V.: Use it or lose it: wear-out and lifetime in future chip multiprocessors. In: Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO). pp. 136–147 (2013)
18. Kim, T.H., Persaud, R., Kim, C.: Silicon odometer: An on-chip reliability monitor for measuring frequency degradation of digital circuits. In: Proceedings of IEEE Symposium on VLSI Circuits. pp. 122–123 (2007)
19. Lu, Z., Huang, W., Stan, M., Skadron, K., Lach, J.: Interconnect lifetime prediction for reliability-aware systems. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 15(2), 159–172 (2007)
20. Mak, T., Cheung, P., Lam, K.P., Luk, W.: Adaptive routing in network-on-chips using a dynamic-programming network. IEEE Transactions on Industrial Electronics 58(8), 3701–3716 (2011)
21. Mak, T., Cheung, P.Y., Luk, W., Lam, K.P.: A dp-network for optimal dynamic routing in network-on-chip. In: Proceedings of the 7th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS). pp. 119–128 (2009)
22. Mercati, P., Bartolini, A., Paterna, F., Rosing, T.S., Benini, L.: Workload and user experience-aware dynamic reliability management in multicore processors. In: Proceedings of the 50th Annual Design Automation Conference (DAC). pp. 1–6 (2013)
23. Mulas, F., Atienza, D., Acquaviva, A., Carta, S., Benini, L., De Micheli, G.: Thermal balancing policy for multiprocessor stream computing platforms. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 28(12), 1870–1882 (2009)
24. Ramachandran, P., Adve, S., Bose, P., Rivers, J.: Metrics for architecture-level lifetime reliability analysis. In: Proceedings of IEEE International Symposium on Performance Analysis of Systems and software. pp. 202–212 (2008)
25. Shi, B., Zhang, Y., Srivastava, A.: Dynamic thermal management under soft thermal constraints. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 21(11), 2045–2054 (2013)
26. Srinivasan, J., Adve, S.V., Bose, P., Rivers, J.A.: The case for lifetime reliability-aware microprocessors. In: Proceedings of the 31st Annual International Symposium on Computer Architecture (ISCA). pp. 276–285 (2004)
27. Srinivasan, J., Adve, S.V., Bose, P., Rivers, J.A.: The impact of technology scaling on lifetime reliability. In: Proceedings of 2004 International Conference on Dependable Systems and Networks. pp. 177–186 (2004)

28. Wang, L., Wang, X., Mak, T.: Dynamic programming-based lifetime aware adaptive routing algorithm for network-on-chip. In: Proceedings of 22nd International Conference on Very Large Scale Integration (VLSI-SoC). pp. 1–6 (2014)
29. Wang, X., Li, Z., Yang, M., Jiang, Y., Daneshtalab, M., Mak, T.: A low cost, high performance dynamic-programming-based adaptive power allocation scheme for many-core architectures in the dark silicon era. In: Proceedings of IEEE 11th Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia). pp. 61–67 (2013)
30. Wang, X., Mak, T., Yang, M., Jiang, Y., Daneshtalab, M., Palesi, M.: On self-tuning networks-on-chip for dynamic network-flow dominance adaptation. In: Proceedings of 2013 Seventh IEEE/ACM International Symposium on Networks on Chip (NoCS). pp. 1–8 (2013)
31. Wang, X., Wang, T., Mak, T., Yang, M., Jiang, Y., Daneshtalab, M.: Fine-grained runtime power budgeting for networks-on-chip. In: Proceedings of 20th Asia and South Pacific Design Automation Conference (ASP-DAC). pp. 160–165 (2015)
32. Woo, S., Ohara, M., Torrie, E., Singh, J., Gupta, A.: The splash-2 programs: characterization and methodological considerations. In: Proceedings of the 22nd Annual International Symposium on Computer Architecture (ISCA). pp. 24–36 (1995)
33. Zhu, C., Gu, Z., Dick, R., Shang, L.: Reliable multiprocessor system-on-chip synthesis. In: Proceedings of the 5th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS). pp. 239–244 (2007)
34. Zhuo, C., Sylvester, D., Blaauw, D.: Process variation and temperature-aware reliability management. In: Proceedings of Design, Automation Test in Europe Conference Exhibition (DATE). pp. 580–585 (2010)
35. Zong, W., Wang, X., Mak, T.: On multicast for dynamic and irregular on-chip networks using dynamic programming method. In: Proceedings of the 6th International Workshop on Network on Chip Architectures (NoCArc). pp. 17–22 (2013)