Journal of Computer Applications
计算机应用,2012,32(3):855 -860
ISSN 1001-9081
CODEN JYIIDU
2012-03-01
http://www.joca.cn

文章编号:1001 -9081(2012)03 -0855 -06
doi:10.3724/SP.J.1087.2012.00855

# 基于π网的嵌入式系统软硬件划分方法

郭荣佐[1*],黄　君[2],王　霖[3]

(1.四川师范大学 计算机科学学院,成都 610068;　2.四川工商职业技术学院 基础部,四川 都江堰 611830;
3.成都纺织高等专科学校 电子信息与电气工程系,成都 611731)

(*通信作者电子邮箱 gyz00001@163.com)

**摘　要**:针对嵌入式系统软硬件划分问题,提出一种基于π网的软硬件划分方法。首先介绍π网的定义及其规则;然后,对嵌入式系统目标进行描述和定义,应用π网建立软硬件划分模型——嵌入式系统软硬件划分模型(ESHPM);最后,对模型ESHPM进行一致性、死锁和兼容性分析,同时,对模型ESHPM进行了优化。所建立的模型ESHPM满足一致性,各进程间无死锁存在,且各个进程之间的交互是兼容的;同时模型ESHPM有效地提高了划分精度,并获得了更加合理的软硬件划分方法。

**关键词**:π网;π演算;嵌入式系统;软硬件划分;优化算法

**中图分类号**:TP311.5　　**文献标志码**:A

# Hardware/software partitioning method of embedded system based on π-nets

GUO Rong-zuo[1*], HUANG Jun[2], WANG Lin[3]

(1. *College of Computer Science, Sichuan Normal University, Chengdu Sichuan* 610068, *China*;
2. *Department of Basic Courses, Sichuan Technology and Business College, Dujiangyan Sichuan* 611830, *China*;
3. *Department of Electronic Information and Electrical Engineering, Chengdu Textile College, Chengdu Sichuan* 611731, *China*)

**Abstract:** Concerning the partitioning problems of the embedded system software and hardware, a method based on π-nets was proposed to partition the software and hardware of the embedded system. This paper gave a brief introduction to the definition and π-nets rules, and then described and defined the target, and established the division Embedded-system Software and Hardware Partition Model (ESHPM) applying the π-nets of software and hardware of embedded system. Finally this paper analyzed the consistency, deadlock and compatibility; at the same time, optimized the ERSHPM. The ESHPM established in this paper satisfied the consistency and no deadlock between the various processes. And the interaction between the various processes was compatible. The ESHPM effectively improves the accuracy of division, and a more reasonable division method of software and hardware has been got.

**Key words:** π-net; π-calculus; embedded system; software/hardware partitioning; optimization algorithm



## 0　引言

随着计算机软硬件技术的飞速发展,嵌入式系统设计者面临越来越苛刻的性能指标和越来越短的面世时间的挑战,这就要求在设计方法上改进软硬件设计模式,采用软硬件协同设计技术。软硬件协同设计技术的核心是软硬件划分,其作用是从软硬件设计空间中,依据系统功能定义,获得一个满足系统时间、成本、功耗、硬件面积和执行时间等多方面要求的趋于最优的实现,其结果直接决定嵌入式系统设计的优劣。

国内外学者从20世纪90年代开始研究软硬件划分方法,Ernst等[1]提出了以模拟退火算法为基础的嵌入式协同合成结构(COSYnthesis for eMbedded Architecture, COSYMA)系统,在该系统中,所有组件首先完全由软件来实现,再逐步由硬件来实现,直至所设计的系统满足所需的时间约束;而Gupta等[2]在多任务合成框架(VULCAN)系统中使用与Ernst等提出的方法正好相反,也就是系统功能首先全部由硬件实现,再逐步用软件实现且满足时间约束;Maciel等[3]在PISH系统中,提出将Petri网模型用于对系统的软硬件性能指标的定性与定量分析,将结果用于软硬件划分的初始分配阶段。这些方法在一定程度上促进了软硬件划分方法的进步和发展,对嵌入式系统的协同设计技术起到了一定的促进作用。当今,嵌入式系统的设计过程中,要求同时对多个系统进行优化,现有的划分算法不能完全满足需求。本文首先简单介绍π网,然后应用π网建立嵌入式系统软硬件划分模型,并对模型进行分析和优化验证。

## 1　π网简介

π网是将π演算和Petri网这两类并发机制进行有效的结合而形成的一类模块化的高级Petri网,π网的主要构成部分是基本π网和π网的复合规则[4]。在此依据参考文献[4],给出π网的定义。

**定义1**　一个π网$N$是一个四元组:$N=(S,T,F,\tau)$。其中:$S$是库所集,$T$是变迁集,使得$S\cap T=\varnothing$;$F\subseteq(S\times T)\cup(T\times S)$是网$N$的弧集;$\forall s\in S$,若$^*s=\varnothing$则称$s$为输入库所,若$s^*=\varnothing$则称$s$为输出库所,否则称为中间库所;$\tau$是定义在$S\cup F\cup T$上的一个属性函数,使得:

1)$\forall s\in S,\tau(s)=(\lambda_s\mid\alpha_s)$。

其中:

**收稿日期**:2011-08-22;**修回日期**:2011-12-08。　**基金项目**:四川省教育厅自然科学重点项目(10ZA008)。

**作者简介**:郭荣佐(1973 -),男,四川开江人,副教授,主要研究方向:嵌入式系统、物联网感知;　黄君(1974 -),女,重庆人,讲师,硕士,主要研究方向:代数学、形式化验证、本体演化;　王霖(1970 -),男,四川遂宁人,副教授,主要研究方向:语义Web、互联网、嵌入式系统。

$\lambda_s$ 是 $s$ 的标号项。$\lambda_s \in \{e,i,x\}$；$e,i,x$ 分别为网的输入、中间和输出库所的标号。$\alpha_s$ 是 $s$ 的状态项。$\alpha_s = \{Id_s, I_s, O_s, R_s\}$，$Id_s = \varnothing$ 或“$\cdot\sigma$”。当 $Id_s = \varnothing$ 时，称 $s$ 是空库所；否则 $s$ 为非空库所。当 $Id_s = \cdot\sigma$ 时，表示库所 $s$ 中存在一个 token，并且在 token 中还附带有通道名集 $N$ 上的一个替换 $\sigma$，这种替换关系将随着 token 在网中的传递而传递。$I_s$、$O_s$ 分别为进程的抽象名多重集、待输出的通道名多重集。$R_s$ 为受限局部名集（其中的元素分别用 $(x)$、$(y)$、$a$ 等来表示，当 $I_s$ 和 $O_s$ 为独点集时，就直接用其中的元素来表示集合）。

2) $\forall t \in T, \tau(t) = (\lambda_\tau \mid \alpha_\tau)$。

其中：

$\lambda_\tau$ 是变迁 $t$ 的标号项。$\lambda_\tau = \theta_\tau\sigma_\tau, \theta_t \in L = \{\alpha u, \bar{\alpha}y, \bar{\alpha}(y), \omega, \varepsilon \mid \alpha, u, y \in N, \varepsilon \notin \{\omega\} \cup N\}$，$\sigma_t$ 是一个替换函数。当 $\theta_t$ 在 $\alpha u$、$\bar{\alpha}y$、$\bar{\alpha}(y)$、$\omega$、$\varepsilon$ 取值时，$t$ 分别称为输入变迁、自由输出变迁、受限输出变迁、通信变迁和匹配变迁；$\forall \theta_t \in \{\alpha u, \bar{\alpha}y, \bar{\alpha}(y)\}, sub(t) = \{\alpha\}$，表示变迁 $t$ 的主名。当变迁 $t$ 激发时，$\alpha u$、$\bar{\alpha}y$、$\bar{\alpha}(y)$ 分别表示在通道 $\alpha$ 上接受通道名 $u$ 并作替换、在通道 $\alpha$ 上输出自由通道名 $y$ 和在通道 $\alpha$ 上输出受限通道名 $y$。若存在一对变迁 $t_1$、$t_2$，使得 $\theta_{t1} = \alpha u, \theta_{t2} = \bar{\alpha}y$（或 $\theta_{t2} = \bar{\alpha}(y)$），则称 $t_1$、$t_2$ 是一对共轭变迁。

$\alpha_\tau$ 是变迁 $t$ 的出现条件项。$\alpha_\tau$ 由可能出现在 $t$ 中的匹配条件和限制条件组成，$\alpha_\tau$ 应满足：

① 当 $\theta_t = \varepsilon$ 时，形如 $\alpha = \beta$ 的匹配条件成立；

② $\forall s \in {}^*t, sub(t) \notin R_s$。

3) $\forall (s,t), (t,s) \in F, \tau((s,t)) = (Id_{(s,t)}, I_{(s,t)}, O_{(s,t)}, R_{(s,t)}), Id_{(s,t)} = Id_s, I_{(s,t)} \subseteq I_s, O_{(s,t)} \subseteq O_s, R_{(s,t)} \subseteq R_s$。其中：$I_{(s,t)}$ 和 $O_{(s,t)}$ 为将要参加变迁 $t$ 的输入和输出通道名的多重集，$R_s$ 为参加变迁的局部受限名集。

依据 π 演算和 Petri 网基本原理，在定义了 π 网之后，还需要定义其激发条件和变迁规则。

**定义 2** 设 $N = (S,T,F,\tau)$ 是一个 π 网，$t$ 是 $N$ 的一个变迁，则 $t$ 在 $N$ 中的激发条件是：

1) $\forall s \in {}^*t, Id_s \neq \varnothing$；

2) $\forall s \in (t^* - {}^*t), Id_s = \varnothing$；

3) 条件 $\alpha_t$ 为真。

**定义 3** 在某 π 网 $N$ 中，标识 $M^*$ 由 $N$ 的库所状态项组成。若将标识 $M^*$ 视为一个函数，即 $\forall s \in S, M^*(s) = \alpha_t$，则网的一个标识 $M^*$ 在一个变迁 $t$ 激发后，$M^*$ 的后继标识 $M^{*\prime}$ 定义为：

$$M^{*\prime}(s) = \begin{cases} \alpha_s - \tau(s,t), & s \in {}^*t - t^* \\ \alpha_s + \tau(t,s), & s \in t^* - {}^*t \\ \alpha_s - \tau(s,t) + \tau(t,s), & s \in t^* \cap {}^*t \\ \alpha_s, & \text{其他} \end{cases}$$

**定义 4** 设 $t_1$ 和 $t_2$ 是 π 网 $N$ 的一对共轭变迁，若存在 $N$ 的两个不相交的子网 $N_1$ 和 $N_2$，使得 $t_1 \in N_1, t_2 \in N_2$，且 $\alpha_{t1}$，$\alpha_{t2}$ 成真，则 $t_1$ 和 $t_2$ 可在 $N$ 中产生一个通信变迁 $t$，并表示为 $t = \langle t_1, t_2 \rangle, \theta_t = \xi$，使得 ${}^*t = {}^*t_1 \cup {}^*t_2, t^* = t_1^* \cup t_2^*$，$\forall s \in {}^*t_1, \tau(s,t) = \tau(s,t_1)$，且具有如下规则：

设 $\lambda_{t1} = ay\{y/x\}$，

① 当 $\theta_{t2} = \bar{a}y$ 时，则 $\theta_t = \xi\{y/x\}$，且 $\forall s \in t_1^*, \tau(t,s) = \tau(t_1,s)$；

② 当 $\theta_{t2} = \bar{a}(y)$，则 $\lambda_t = \xi\{y/x\}$，且 $\forall s \in t_1^*, \tau(t,s) = \tau(t_1,s) + (\varphi,\varphi,\varphi,\{y\})$。

任意一个 π 网都是由基本 π 网复合规则复合而成。基本 π 网有 4 类，分别称为 Tau 网 $N_\tau$、匹配网 $N_{[a=b]}$、输入网 $N_{\alpha(x)}$ 和自由输出网 $N_{\alpha y}$。依据 π 演算相关理论，得出 π 网的复合规则有并行“$\parallel$”、提升“!”、顺序“;”、选择“[]”和限制“rs”，在此不对这些复合规则予以定义和形式化描述。

## 2 目标定义

在进行嵌入式系统软硬件划分之前，先定义目标系统的模型。

在需求分析阶段给出各个性能指标约束为：性能指标 $iop_1$ 不超过 $B_1$，性能 $iop_2$ 不能超过 $B_2$，……，性能指标 $iop_n$ 不能超过 $B_n$；而同一性能指标能够用硬件实现，也能用软件实现，软件和硬件又拥有各自不同的性能指标数据[5]。

**定义 5** 设 $C$ 为嵌入式系统设计中的一个 IP 核或软件构件，则 $C$ 为一个 $n$ 元组，即 $C = (iop_1, iop_2, \cdots, iop_n)$，其中 $iop_1$，$iop_2, \cdots, iop_n$ 是与 IP 核或软件构件对应的嵌入式系统设计中需要重点考虑和评价的性能参数。

而各种性能指标和约束条件，在进行嵌入式系统设计时必须全面考虑，必要时需进行定性或定量分析与计算。

假设嵌入式系统的功耗为 $P_{total}$，表示系统总的功耗。系统功耗主要由正常运行功耗和空闲功耗两部分构成，空闲功耗是多个模块空闲功耗的累加，运行功耗则由各个功能模块硬件功耗与软件功耗之和的累加[6]，即：

$$P_{total} = P_{free} + P_{run} = \sum_{i \in system} P_{free}(i) + \sum_{\substack{i \in system \\ task \in SWtask(system)}} P(i,task)$$

嵌入式系统成本是所有嵌入式系统开发与设计必须考虑的问题。系统成本主要由开发成本和设备成本构成，而设备成本即为组成系统所需的 IP 核固件及软件构架等决定，因此系统成本就为各个功能模块软硬件成本之和，即：

$$C_{castSys}(system) = \sum_{i \in system} C_i$$

系统的体积、面积和重量等，也是需要考虑的指标。但相比之下，对系统的执行时间而言，系统执行时间就显得尤为重要，特别是嵌入式实时系统中，执行时间是主要的指标参数。嵌入式系统的执行时间 $T_{system}$ 由硬件执行时间和软件执行构成[7]，即：

$$T_{system} = T_{SW} + T_{HW} = \sum_{i \in HW(system)} T_{HW}(i) + \sum_{\substack{i \in HW(system) \\ task \in SWtask(system)}} T(i,task)$$

假设嵌入式系统的各性能参数指标分别为：系统功耗 $\leqslant P$；系统成本 $\leqslant CC$；面积 $\leqslant A$；重量 $\leqslant W$；体积 $\leqslant V$；执行时间 $\leqslant T$；则定义 5 所定义的 $C$ 表示为：

$$C = (P_{ik_0}, CC_{ik_1}, A_{ik_2}, W_{ik_3}, V_{ik_4}, T_{ik_5})$$

其中：$i$ 表示某个 IP 核或软件组件，且 $1 \leqslant i \leqslant n$；$k_0, k_1, k_2, k_3, k_4, k_5$ 为确定的常数，表示功能模块的备选项的数目。

**定义 6** 设嵌入式系统由若干功能单元组成，即：$E_{SFS} = \{Func_1, Func_2, \cdots, Func_M\}$，则每个功能单元对应一个 IP 核或软件构件备选集合，即：

$Func_1 \rightarrow \{C_{11}, C_{12}, \cdots, C_{1a}\}$

$Func_2 \rightarrow \{C_{21}, C_{22}, \cdots, C_{2b}\}$

$\cdots$

$Func_M \rightarrow \{C_{m1}, C_{m2}, \cdots, C_{mn}\}$

其中:$C$ 为服从定义 5;$a,b,\cdots,n$ 是常数,分别表示 $Func_1$,$Func_2,\cdots,Func_M$ 所对应的备选 IP 核或软件构件的数目。

因此,嵌入式系统软硬件划分问题就转化为求 $E_{SFS}$ 的一个组合$(C_{1i},C_{2j},\cdots,C_{nt})$,其中 $1 \leqslant i \leqslant k_0, 1 \leqslant j \leqslant k_1,\cdots,1 \leqslant t \leqslant k_5$,且满足:

$$P_{1i} + P_{2j} + \cdots + P_{nt} \leqslant P$$

$$CC_{1i} + CC_{2j} + \cdots + CC_{nt} \leqslant CC$$

$$\cdots$$

$$T_{1i} + T_{2j} + \cdots + T_{nt} \leqslant T$$

因此,可得所定义的目标嵌入式系统如图 1 所示。

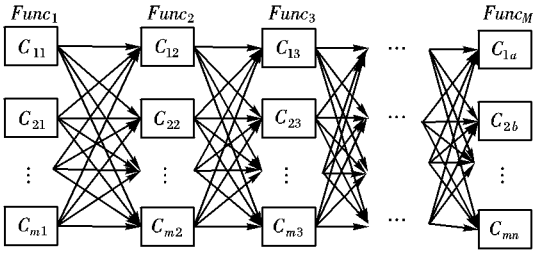


图 1　嵌入式系统目标结构

## 3　软硬件划分模型建立

### 3.1　模型建立

依据 π 网基本原理和目标系统的定义,可以形式化定义嵌入式系统软硬件划分模型(Embedded-system Software and Hardware Partition Model, ESHPM)。

**定义 7**　嵌入式系统软硬件划分模型 ESHPM 为一个 π 网 $N_{ESHPM}$,即 $N_{ESHPM} = (E_{HW},E_{SW},E_F;E_\tau)$,其中:$E_{HW}$ 为系统设计中的硬件实现的功能集合,是网 $N_{ESHPM}$ 的库所集;$E_{SW}$ 为系统设计中软件实现的功能集合,是网 $N_{ESHPM}$ 的变迁集,使得 $E_{HW} \cap E_{SW} = \varnothing$;$E_F \subseteq (E_{HW} \times E_{SW}) \cup (E_{SW} \times E_{HW})$ 是网 $N_{ESHPM}$ 的有向弧集,表示网 $N_{ESHPM}$ 的对应关系;$E_\tau$ 是定义在 $E_{HW} \cup E_{SW} \cup E_F$ 上的一个属性函数,表示嵌入式系统设计中的各种约束评价因子所构成的函数关系。

由定义 7 可知,嵌入式系统软硬件划分就是在系统功能集合上 $E_{SFS} = \{Func_1,Func_2,\cdots,Func_M\}$ 做一个合理的划分,即 $E_{SFS} = \{E_{HW},E_{SW}\}$,$E_{HW} \cup E_{SW} = E_{SFS}$,$E_{HW} \cap E_{SW} = \varnothing$,$\forall e_{HW} \in E_{SFS}$,则 $e_{HW} \in E_{HW}$ 或 $e_{SW} \in E_{SW}$,并且在保证 $E_F$ 和 $E_\tau$ 条件下使得代价最小。

**定义 8**　约束评价因子函数 $E_\tau$ 为一个 8 元组,以表示各种评价因子间的相互关系[8],即:$E_\tau = (P,T,Q,F_C,F_D,W,D,I)$。

其中:

$P = \{P_1,P_2,\cdots,P_n\}$ 是有限的库所集,在此表示各种功能部件的功耗,包括硬件实现和软件实现的功能的功耗;

$T = \{t_1,t_2,\cdots,t_n\}$ 为有限迁移集合$(P \cup T \neq \varnothing, P \cap T = \varnothing)$ 为表示由硬件、软件实现的功能部件执行时间的集合,反映嵌入式系统的动态迁移;

$Q = \{q_1,q_2,\cdots,q_h\}$ 是有限的数据变迁集,且 $P \cap T = \varnothing, P \cap Q = \varnothing, Q \cap T = \varnothing$,在此表示各功能部件的成本;

$F_C \subseteq (P \times T) \cup (T \times P) \cup (P \times Q)$ 是控制流的弧的集合,$F_D \subseteq (P \times Q) \cup (Q \times P) \cup (P \times T)$ 是数据流的弧的集合,表示硬件实现与软件实现之间的动态关系;

$W: P \to X$(非负整数集合)是位置集合上的标识的权,表示各种约束评价因子的优先级别;

$D: T \to T'$ 为迁移所需的时间,表示网 $N_{ESHPM}$ 迁移为 $N'_{ESHPM}$ 所需的时间;

$I$:表示嵌入式系统设计中的其他评价因子,如系统面积、可靠性和安全性等。

定义 7 ~ 8 定义了嵌入式系统软硬件划分的 ESHPM,还需定义 ESHPM 的动态迁移规则。

规则 1　网 $N_{ESHPM}$ 的进程标识 $M_{Rm,i}(i = 1,2,\cdots,n)$,迁移使能规则如下。

1) 数据迁移使能规则:

$\forall q \in Q, \exists p_i \in {}^*q$,使 $M_c(p_i) \geqslant W_C(p_i,t)$,则 $q$ 在当前标识下可使能;

2) 控制迁移使能规则:

$\forall q \in Q, \exists p_i \in {}^*q$,如果 $M_c(p_i) \geqslant W_C(p_i,t)$,且 $\forall p_i \in {}^\circ t, D(M_D(p_i) \in {}^\circ t) = 1$,则 $t$ 在当前标识下可使能。

规则 2　网 $N_{ESHPM}$ 的进程标识 $M_i(i = 1,2,\cdots,n)$,使能迁移引发标识变化规则如下。

1) 数据变迁引发标识变化规则:

$\forall p_i \in {}^\circ q, p_j \in q^\circ$,则 $M^{k+1}(p_j) = \sum_i M^k(p_i) \cdot W_D(p_i,q)$

2) 控制变迁引发标识变化规则:

如果令牌移出,则 $\forall p_i \in {}^*t, M_c^{k+1}(p_i) = M_c^k(p_i) - W_c(p_i,t)$;

如果令牌加入,则 $\forall p_i \in t^*, M_c^{k+1}(p_i) = M_c^k(p_i) + W_c(t,p_i)$。

由此可见数据变迁实现对数据的操作控制变迁在判断条件成立时实现控制判断。

在此,为简化对 ESHPM 的描述,仅将 ESHPM 中的迁移看成是操作,而位置为操作的输入或输出,即:若 $e_s \in E_{SFS}$,则 $e_s$ 为一个操作,对于 $\forall P_i \in {}^*e_s$,则 $P_i$ 为操作的输入;对于 $\forall P_j \in e_s^*$,则 $P_j$ 为操作 $e_s$ 的输出。

利用基于 Petri 网的可执行的规格说明语言 Exspect 及其软件和前面定义,对模型进行描述和实现[9]。应用 Exspect 结构化描述语言,对模型的类型、函数、过程和系统进行定义,对软硬件层次进行分解,对功能相同的软硬件模块进行分类描述和封装,然后给每个功能模块以相应的参数,可得出 π 网的迁移变迁与目标系统软硬件划分结果的映射关系,如图 2 所示。

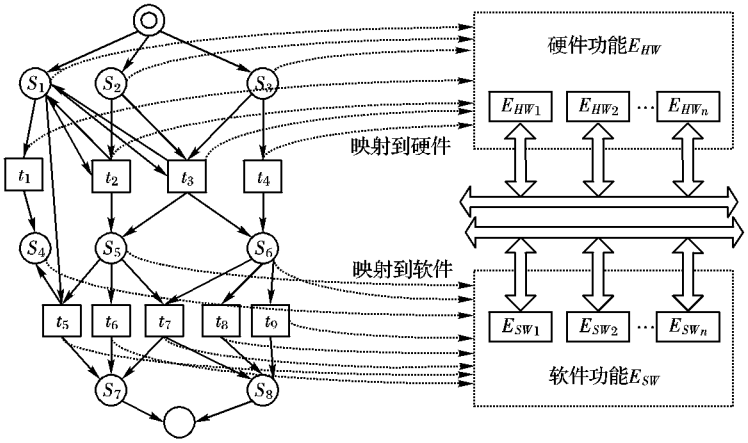


图 2　π 网的迁移变迁与划分结果的映射关系

### 3.2 模型变迁时间度量

$N_{ESHPM}$ 网中 $E_\tau$ 被定义为嵌入式系统设计中的约束评价因子，依据定义 8 可知，$E_\tau$ 的分量 $D$ 表示网 $N_{ESHPM}$ 的变迁时间，该时间是软硬划分中的一个十分重要的因子。它是性能优化和确认时间约束所必需的。可以使用 π 网的相关理论对 $N_{ESHPM}$ 的变迁时间进行度量，以确定嵌入式系统软硬件划分方法的执行时间，需先给出相关概念[4]。

**定义 9** 设 $N_1$、$N_2$ 是两个 π 网，即 $N_1 = (S_1, T_1, F_1, \tau_1)$，$N_2 = (S_2, T_2, F_2, \tau_2)$。若 $\forall f: S_1 \cup T_1 \to S_2 \cup T_2$ 满足：

①$f(S_1) = S_2$，$f(T_1) = T_2$，$f^{-1}(S_2) = S_1$，$f^{-1}(T_2) = T_1$；

②$F_1 = F_2$，即 $\forall s \in S_1, t \in T_1$，$f(^*s) = \{f(t) \mid t \in {}^*s\} = {}^*f(s)$，$f(s^*) = \{f(t) \mid t \in s^*\} = f(s)^*$，$f(^*t) = \{f(s) \mid s \in {}^*t\} = {}^*f(t)$，$f(t^*) = \{f(s) \mid s \in t^*\} = f(t)^*$；

③$\tau_1(s) = \tau_2(f(s))$，$\tau_1(t) = \tau_2(f(t))$；

④$\forall s \in S_1, Id_s = \varnothing, t \in s^*, \tau_1((s,t)) = \tau_2((f(s), f(t)))$。

则称 $N_1$ 和 $N_2$ 是同构的，记为 $N_1 \cong N_2$，$f$ 称为 $N_1$ 和 $N_2$ 的同构映射。

**定义 10** 若 $N_1$、$N_2$ 是同构的 π 网，且满足 $\tau_1(s,t) = \tau_2(f(s), f(t))$，$\tau_1(t,s) = \tau_2(f(t), f(s))$，则称 $N_1$ 和 $N_2$ 是恒等的，$f$ 称为 $N_1$ 和 $N_2$ 的恒等映射[4]，记为 $N_1 \equiv N_2$。

**定义 11** 设 $N_1$、$N_2$ 是两个 π 网，$\propto$ 是由所有 π 网构成的集合 $P_N$ 上的一个二元关系。若 $N_1 \propto N_2$，当且仅当存在 $N_1$ 的某个 $\alpha$ 转换 $N_\alpha$ 时，使得 $N_1 \equiv N_2 \parallel Z, A \cap f_n(N_1) = \varnothing, Z$ 是 $n$ 个零网的并行复合，$n \in Z^+$，$n \geqslant 0$，$f_n(N_1)$ 是网 $N_1$ 的自由名集合；则称 $N_1$、$N_2$ 是同余的，记为 $\propto$。

$N_{ESHPM}$ 网经过变迁和运算，首先求得 $N_{ESHPM}$ 网的同构网 $N'_{ESHPM}$；将同构网 $N'_{ESHPM}$ 经过 π 网的 INPUT、OUTPUT、MATCH、COM 和 CLOSE 等规则运算，得到 $N_{ESHPM}$ 网的恒等网 $N''_{ESHPM}$；最后将 $N''_{ESHPM}$ 网与 $N'_{ESHPM}$ 网和 $N_{ESHPM}$ 一起求其同余网 $N''''_{ESHPM}$，即得到最终目标 π 网。其中，$N_{ESHPM}$ 网是经过迁移到 $N''''_{ESHPM}$ 网，因此，可以看成是系统变迁的一条路径，根据 $D: T \to T'$ 对其计算运行时间，其中的最大值所对应的路径就是关键路径，所需时间即为关键路径时间。若给定了一种划分方案，每一个迁移的时间也就随之确定，根据以上算法可以对这种划分方案的关键路径时间进行计算，从而可以计算出该方案的执行时间性能参数。

## 4 模型分析与优化

π 网是 Petri 网和 π 演算有效结合而成的一种高级 Petri 网，在此可用 π 演算来验证 ESHPM。

### 4.1 一致性分析

用 π 演算的行为等价理论，来判定两个功能相同软硬件进程在并发执行、并与环境交互的进程是否具有相等行为。如果两个进程行为相等，那么从系统环境中取出其中一个而放入另外一个，系统不会感知替换发生[10]。若软件进程 $P$ 和硬件进程 $Q$ 是一致的，则 $P$ 和 $Q$ 至少能借助一个共享的对偶名字进行交互，$P$ 和 $Q$ 之间存在同步关系。

**定义 12** 设同步进程集合上的二元关系 $R_f$，对于 $PR_fQ$，且对所有的替代 $\sigma \notin f_n(P) \cup f_n(Q)$，$f_n$ 为进程的自由名集合，满足 $P\sigma RQ\sigma$，而且下列条件成立：

1）如果 $P \xrightarrow{\tau} P'$，则 $P'R_fQ$；

2）如果 $P \xrightarrow{x(w)} P' \wedge Q \xrightarrow{\bar{x}y} Q'$，则 $P'(y/w)R_fQ'$；

3）如果 $P \xrightarrow{x(w)} P' \wedge Q \xrightarrow{\bar{x}(w)} Q'$，则 $P'C_fQ'$。

则称 $R_f$ 为进程间弱一致性关系，又称半一致性关系。如果 $R$ 和 $R^{-1}$ 都是弱一致性关系，则称 $R$ 为进程间半一致性关系[9]，记为 $\backsim$。

**定义 13** 如果 $R_f$ 和 $R'_f$ 都是半一致性关系，则称 $R_f$ 为一致性关系，如果 $PR_fQ$，则称进程 $P$、$Q$ 具有一致性[9]，记为 $P \backsimeq Q$。

进程的一致性确保在进程交互中不存在不匹配的情况，强调进程的正常交互，表明进程能够执行内部演化而成功结束[11]。

**定理 1** 设 $P$ 和 $Q$ 为半一致性进程 $P \backsim Q$，如果构件 $C_{SW,1}$ 和 $C_{HW,1}$ 能分别由 $P$ 和 $Q$ 正确描述，则构件 $C_{SW,1}$ 和 $C_{HW,1}$ 满足半一致性。

**定理 2** 设 $P$ 和 $Q$ 为一致性进程 $P \backsimeq Q$，如果构件 $C_{SW,1}$ 和 $C_{HW,1}$ 能分别由 $P$ 和 $Q$ 完整而正确的描述，则构件 $C_{SW,1}$ 和 $C_{HW,1}$ 满足一致性，即构件 $C_{SW,1}$ 和 $C_{HW,1}$ 能正常交互。

**定理 3** 如果模型 ESHPM 完全满足一致性，则该模型可以正常交互。

由上分析可知，ESHPM 满足一致性要求，各个进程之间能够正常交互。

### 4.2 死锁分析

局部的一致性分析并不能确保模型 ESHPM 无死锁性，还要利用 π 演算的相关分析方法对 ESHPM 进行死锁分析。π 演算行为等价理论指出，若两个进程行为等价，则在交互过程中，用一个进程替换另一个进程，都不能从外部发现产生了差别从而能够判定发生了替换[12]。在进行构件替换、端口角色(Role)连接时，通常两者的行为并不完全等价。

**定义 14** 设 π 演算进程 $P$，存在进程 $P'$，若

$$P \Rightarrow P' \wedge \{P' \to P''\} = \varnothing \wedge P' \equiv 0$$

成立，则进程 $P$ 不存在死锁。进程 $P$ 不存在死锁，表明进程 $P$ 能够内部演化到进程 $P'$，而且能完全执行内部演化，即 $P$ 结构等同于进程 0。进程不存在死锁，表明该进程能够内部演化而成功结束。又由于进程 $P$ 与 $Q$ 具有一致性，则 $P \mid Q$ 不存在死锁。

在 ESHPM 运行过程中，构件替换、端口角色连接时，通常两者的行为并不完全等价，而是可能不同，也就是不能完全满足一致性要求，因此，不能完全确定系统不会发生死锁。

**定义 15** 设进程集合之上的二元关系 $R_f$，如果 $PR_fQ$，则如下条件成立：

1）如果 $P \xrightarrow{x(y)} P'$，则 $\exists Q': Q \xrightarrow{x(y)} Q' \wedge P'R_fQ'$；

2）如果 $P \Rightarrow P'$，则 $\exists Q': Q \Rightarrow Q' \wedge P'R_fQ'$；

3）如果 $P \overset{\overline{x_1y_1}}{\Longrightarrow} P_1, P \overset{\overline{x_2y_2}}{\Longrightarrow} P_2, \cdots, P \overset{\overline{x_ny_n}}{\Longrightarrow} P_n$，则 $\exists Q', \bar{x_i}y_i$ $(1 \leqslant i \leqslant n): Q \overset{\overline{x_iy_i}}{\Longrightarrow} Q' \wedge P_iR_fQ'$；

4）如果 $P \overset{x_1(y_1)}{\Longrightarrow} P_1, P \overset{x_2(y_2)}{\Longrightarrow} P_2, \cdots, P \overset{x_n(y_n)}{\Longrightarrow} P_n$，则 $\exists Q', \bar{x_i}y_i$ $(1 \leqslant i \leqslant n): Q \overset{x_i(y_i)}{\Longrightarrow} Q' \wedge P_iR_fQ'$。

则称二元关系 $R_f$ 为半替换关系，用符号“$<$”表示。若 $R_f$ 为半替换关系，即 $PR_fQ$ 成立，则称 $Q$ 可半替换 $P$，记为 $P < Q$。

**定义 16** 设进程集合之上的二元关系 $R_f$，如果 $R_f$、$R'_f$ 都是半替换关系，则称关系 $R_f$ 为替换关系。如果 $PR_fQ$，则称 $P$ 和

$Q$ 可相互替换,记为 $P <> Q$。

**定理 4**　如果进程 $P$ 和 $Q$ 满足 $P <> Q$,则进程 $P$ 和 $Q$ 可以相互替换,而系统不能察觉。

**定理 5**　如果系统中任意两进程 $P$ 和 $Q$ 满足 $P <> Q$,则系统不存在死锁。

由上述定义、定理可知,ESHPM 的进程不存在死锁,而且进程间不存在死锁。

### 4.3　兼容性分析

由前面内容可知 ESHPM 符合 π 演算的一切规则,因此可以用 π 演算来验证 ESHPM 的兼容性。

将嵌入式系统软硬件划分表达成 π 演算进程,将软硬件之间的转换表达成进程表达式,验证 ESHPM 的兼容性就可转化成对进程表达式的演算。而软硬件之间转换可以理解为软件组件提供的功能与硬件 IP 核提供的功能能够互换,因此将这种互换定义为服务。文献[13]给出了在 π 演算中两个服务是否兼容的判定定理:给定两个服务 $s_1$ 和 $s_2$ 以及两个服务对应的进程表达式 $P_{s1}$ 和 $P_{s2}$,若这两个服务是可兼容的,则这两个服务的进程必定满足条件 $P_{s1} \mid P_{s2} \Rightarrow 0$。

给定服务 $s_i$ 及其对应的操作集合 $Opr(s_i)$,$s_i$ 所做操作为一个有序队列 $OprQ\langle Opr_e,\cdots Opr_f,\cdots,Opr_g\rangle$,即从 $s_i$ 的起始状态 $Status_0$ 迁移到任意一个状态 $Status'$ 所做操作为有序队列称为服务 $s_i$ 的一条操作序列。若 $Status'$ 为服务 $s_i$ 的结束状态,则称该操作序列为服务 $s_i$ 完全操作序列。若组合服务集合 $S_n$,则 $S_n$ 中的 $n$ 个服务之间的交互因子可表示为一个三元组 $\theta = (opr_{out}, opr_{in}, H)$,其中 $opr_{out}$ 表示服务 $s_i$ 的输出操作集合 $Opr_{out}(s_i)$,$opr_{in}$ 表示服务 $s_i$ 的输入操作集合 $Opr_{in}(s_i)$ 或 $Opr_{in}(s_i) = \varnothing$,$H$ 是输入操作集合的函数即 $H = f_h(opr_{in})$ 且满足 $H \subseteq f_h(opr_{out})$。由此可见,交互因子是服务在交互过程中的一个交互步骤,一个服务输出消息后,由组合中另一个服务得到该消息,对另一服务而言,就是输入信息[14]。其中,$opr_{in}$ 为 $\varphi$ 表示没有服务接收消息,交互失败,因此,交互因子既能描述服务之间交互成功,又能表示服务之间交互失败。

如果组合服务集合 $S_n$ 以及 $S_n$ 中所有服务的一个完全操作序列所组成的集合 $Opr_s$,那么 $S_n$ 中的 $n$ 个服务之间的一条交互路径为一个交互因子有序队列 $OprQs = \langle\theta_1,\theta_2,\cdots\rangle$,且按照以下步骤构造 $OprQ$:① 令 $OprQ = \varnothing$;② 若 $Opr_{outk}$ 是 $Opr_s$ 中操作序列 $OprQ_i$ 的第一个操作,且 $Opr_{ink}$ 是操作序列 $OprQ_j$ 的第一个操作,$i \neq j$,则将 $\theta_k$ 放在 $OprQs$ 的最后面;③ 从 $OprQs$ 中删除 $Opr_{outk}$ 和 $Opr_{ink}$;④ 重复 ② ~ ③,直到不能从 $OprQs$ 中删除任何操作为止。由此保证了 $S_n$ 中的服务都是从起始状态开始进行交互,直至交互结束或交互无法再继续下去。将 $OprQs = \langle\theta_1,\theta_2,\cdots\rangle$ 一直进行上述操作后,若 $OprQs = \varnothing$,则 $OprQs$ 为有效交互路径。

**定义 17**　若集合 $S_n$ 的 $n$ 个服务之间存在一条有效交互路径,则称这 $n$ 个服务是可兼容的。

**定理 6**　给定集合 $S_n$ 的 $n$ 个服务 $s_1,s_2,\cdots,s_n$ 以及 $n$ 个服务对应的进程 $P_1,P_2,\cdots,P_n$,如果这 $n$ 个服务是可兼容的,则这 $n$ 个服务的进程必定满足条件 $P_1 \mid P_2 \mid \cdots \mid P_n \Rightarrow 0$。

对 ESHPM 中的所有构件,无论是软件组件还是硬件 IP 核,都用 π 演算的进程予以描述,并应用上面的方法,便可得到 $OprQ_{ESHPM} = \varnothing$,从而得到模型的所有服务是存在一条有效交互路径,即所有服务是可兼容的。根据定义 17 和定理 6 可知,ESHPM 具有兼容性。

### 4.4　模型优化

嵌入式系统软硬件划分的目标是搜索功能分配到硬件或软件模块,使设计不仅能满足系统执行时间限制,而且使成本、功耗、面积、重量和体积等达到最优。经过比较,本文采用进化多目标优化算法(Evolutionary Multi-objective Optimization Algorithms, EMOA)实现对 $N_{ESHPM}$ 的优化。

依据参考文献[15 - 17]提出的理论,进化算法是通过在代与代之间维持由潜在解组成的种群来实现全局搜索,这种从种群到种群的方法对于搜索多目标优化问题的 Pareto 最优解集是很有用的。因此,依据参考文献,设计的优化算法流程如图 3 所示,其优化算法步骤如下:

第 1 步　进行 Pareto 排序。采用 DBLF 调度算法对任务进行调度,估算系统成本与功耗;Pareto 排序。

第 2 步　计算个体共享度、个体适应度赋值。

第 3 步　进行选择、交叉和变异运算。

第 4 步　进行父代与子代混合组成新种群,根据等级和适应度值,取前 50% 作为下一代种群。

第 5 步　输出当前种群中的 Pareto 最优可行解。

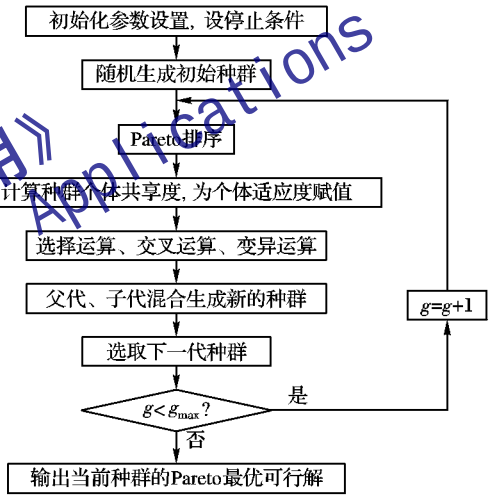


图 3　EMOA 算法流程

以 π 网性能评估方法为基础,对 EMOA、遗传算法(Genetic Algorithm, GA)和禁忌搜索(Tabu Search, TS)算法做了对比试验,输入为同一个迁移数的 π 网[13]。由于控制参数对实验结果有一定的影响,因此,在实验中使用了相同的控制参数,以保证可比性。经过实验,得到如图 4 所示的实验结果。经过分析比较,得到 EMOA 算法具有较强的划分能力,特别适合运用于复杂且对划分精度要求高的嵌入式系统。

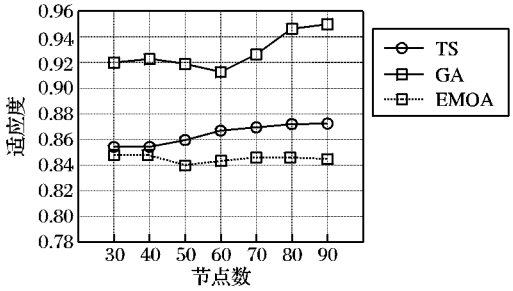


图 4　适应度/节点规模曲线

## 5　结语

本文应用参考文献[4]提出的 π 网基本原理和方法,对嵌入式系统软硬件划分方法进行研究,提出了基于 π 网的嵌

入式系统软硬件划分模型(ESHPM)。通过对比实验结果可知,本文设计的嵌入式系统软硬件划分模型的性能是较优越的。本文所建立的 ESHPM 满足一致性,各进程间无死锁存在,且各个进程之间的交互是兼容的;同时 ESHPM 有效地提高了划分精度,并获得了更加合理的软硬件划分方法。

**参考文献:**


[1] ERNST R, HENKEL J, BENNER T. Hardware-software cosynthesis for microcontrollers[J]. IEEE Design & Test of Computers, 1993, 10(4): 64 - 75.

[2] GUPTA R K, COELHO C, de MICHELI G. Synthesis and simulation of digital systems containing interacting hardware and software components[C]// DAC'92: Proceedings of the 29th ACM/IEEE Design Automation Conference. Los Alamitos, CA: IEEE Computer Society Press, 1992: 225 - 230.

[3] MACIEL P, BARROS E, ROSENSTIEL W. A Petri net model for hardware software codesign[J]. Design Automation for Embedded Systems, 1999, 4(10): 243 - 310.

[4] 曹木亮. 基于 π-演算的 Petri 网和密码协议的形式化分析[D]. 上海: 上海交通大学, 2007: 20 - 75.

[5] LEILA S, AUGUSTO S, EDNA B. A constructive approach to hardware/software partitioning[J]. Formal Methods in System Design, 2004, 24(1): 45 - 90.

[6] 徐海涛. 基于 SOPC 的软硬件划分算法研究[D]. 哈尔滨: 哈尔滨理工大学, 2009: 48 - 52.

[7] 李兰英, 冯宏伟. 基于多性能指标的 SoC 软硬件划分方法研究[J]. 计算机工程与应用, 2008, 44(12): 126 - 129.

[8] 郭荣佐, 郭进, 王霖. 嵌入式系统软件体系结构动态建模及应用研究[J]. 计算机应用, 2009, 29(4): 1153 - 1158.

[9] 曲长征, 于永利, 金伟, 等. 基于 ExSpect 的复杂离散事件动态系统建模[J]. 系统仿真学报, 2005, 17(12): 3011 - 3013.

[10] MILNER R, PARROW J, WALKER D. A calculus of mobile processes Pt. 2[J]. Journal of Information and Computation, 1992, 100(1): 41 - 77.

[11] MILNER R. Communicating and mobile systermsahe π-calculus [M]. Cambridge: Cambridge University Press, 1999: 20 - 30.

[12] SANGIORGI D, WALKER D. The π-calculus: A theory of mobile process[M]. Cambridge: Cambridge University Press, 2001: 32 - 42.

[13] 邓水光. Web 服务自动组合与形式化验证的研究[D]. 杭州: 浙江大学, 2007: 93 - 116.

[14] 贾志淳, 陈荣, 张维石. 航空订票业务的 Web 服务建模及组合兼容性验证[J]. 计算机工程与应用, 2010, 46(24): 237 - 242.

[15] 李康顺, 李元香, 康立山, 等. 一种基于输运理论的多目标演化算法[J]. 计算机学报, 2007, 30(5): 797 - 805.

[16] 彭春华, 孙惠娟, 郭剑峰. 求解 PMU 多目标优化配置问题的非劣排序微分进化算法[J]. 控制理论与应用, 2009, 26(10): 1075 - 180.

[17] 钱顺, 蔡兰英, 石敏, 等. 基于遗传和禁忌搜索混合的软硬件划分算法[J]. 计算机工程与应用, 2009, 45(20): 81 - 84.


(上接第 854 页)

表 2 消防站覆盖指标对比 %

| 选址方式 | 覆盖率 | 交叉覆盖率 |
|---|---|---|
| 离散 | 79.91 | 38.3 |
| 连续 | 89.97 | 24.3 |

## 4 结语

本文提出连续覆盖的模型,将模拟退火算法加以改进并使用 Matlab 语言,以消防站在有效响应时间内的覆盖面积最大为目标函数,进行编程,在退火过程中通过控制降温参数来实现基于连续空间的消防站的选址优化。通过实例应用研究,得到以下结论:

1)基于连续选址的解空间突破了图二元结构的束缚,实现了空间内任意坐标的选址和精确的覆盖,覆盖效果大为提高;

2)模拟退火算法能有效地应用于大规模城市空间的消防站选址问题的研究,算法简单并能得到较稳定的全局最优解,实用性强,效率高;

3)连续选址消防站分布比较均匀,且在人口稠密区消防站密度也较大,比较符合实际需求。

消防站选址布局规划除考虑出警时间这一定量因素外,还与土地资源、土地征用费、环境气候等因素相关,需对消防站选址进行动态的修正和调整,并根据行政区域进一步划分各消防站的辖区范围,以实现消防资源的优化。考虑更多因素约束的消防站选址问题,以期能更好地符合实际应用的需求,将是今后需要进一步研究的问题。

**参考文献:**


[1] HELLY W. Urban systems models [M]. New York: Academic Press, 1975.

[2] MASOOD B, AMRK M. A multi-objective model for locating fire station [J]. European Journal of Operational Research, 1998, 110(2): 243 - 260.

[3] 胡运权. 运筹学教程[M]. 北京: 清华大学出版社, 1990.

[4] 何晋强, 黎夏, 刘小平, 等. 蚁群智能及其在大区域基础设施选址中的应用[J]. 遥感学报, 2009, 13(2): 246 - 256.

[5] 陈慕齐, 陈迎春, 齐欢. 基于混合遗传算法的试验选址问题研究[J]. 武汉理工大学学报, 2006, 30(5): 877 - 880.

[6] YANG LILI, JONES B F, YANG SHUANG-HUA. A fuzzy multi-objective programming for optimization of fire station locations through genetic algorithms [J]. European Journal of Operational Research, 2007, 181(2): 903 - 915.

[7] 姚泽清, 苏晓冰. 应用泛函分析[M]. 北京: 科学出版社, 2008.

[8] 王凌. 智能优化算法及其运用[M]. 北京: 清华大学出版社, 2001.

[9] 李香平, 张红阳. 模拟退火算法原理及改进[J]. 软件学报, 2008, 7(4): 47 - 48.

[10] 何寿奎. 城市消防站点布局的改进启发式算法[J]. 数学的实践与认知, 2008, 40(1): 143 - 146.

[11] 吴军. 消防站优化布局方法与技术[J]. 消防科学与技术, 2006, 25(1): 100 - 102.

[12] 陈驰, 任爱珠. 消防站布局优化的计算机方法[J]. 清华大学学报: 自然科学版, 2003, 43(10): 1390 - 1393.

[13] 马云峰, 杨超, 张敏, 等. 基于时间满意的最大覆盖选址问题[J]. 中国管理科学, 2006, 14(2): 45 - 51.