AUTOSAR可运行实体-任务自动映射方法研究

冉 正 罗 蕾 晏 华 李 允
(电子科技大学计算机科学与工程学院 成都 611731)

摘 要 下一代汽车电子标准AUTOSAR定义汽车应用程序设计过程包括系统级设计和ECU级设计。系统级设计以软件构件为单位来设计应用,其中软件构件包含一组可运行实体。ECU级设计主要将可运行实体代码组织为嵌入式实时操作系统任务。因此,在将分配到ECU的软件构件集转换为实时系统任务集的过程中,需要有经验的嵌入式开发工程师进行可运行实体-任务的映射配置,以保证系统的实时性。鉴于可运行实体-任务的映射配置工作具有配置需求量大、复杂度高等特点,文中设计了一种可运行实体-任务自动映射方法。该方法综合考虑了可运行实体的触发关系、周期需求、数据共享等因素,对提高汽车软件开发效率具有非常重要的实用价值。最后,将该方法应用于AUTOSAR标准的汽车电子巡航控制系统实例中。实验结果显示,所提方法在抖动时间、阻塞时间、调度频繁度和数据通信量4个方面都具有良好的表现。

关键词 汽车电子,ECU配置,可运行实体,任务,映射
中图法分类号 TP37 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.04.032

Study on Automatic Method for AUTOSAR Runnable Entity-task Mapping

RAN Zheng LUO Lei YAN Hua LI Yun
(School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China)

Abstract The next generation automotive electronic standard AUTOSAR defines that the automotive application design process includes system level design and ECU level design. Software components are function units of application in system level design and each software component comprises a set of runnable entities. The main task of ECU level design is organizing the code segments of runnable entities as embedded operating system tasks. In the process of transforming the component set which is assigned from one ECU into a real-time system task set, the experienced embedded development engineers are necessary for runnable entity-task mapping configuration to ensure the real-time performance of the system. As the requirements of runnable entity-task mapping configuration are large and complex, this paper proposed a runnable entity-task automatic mapping method. With the consideration of trigger relationship between runnable entities, period requirements, data sharing and other factors, this method has important practical significance in improving the efficiency of automotive software development. Finally, the proposed method was applied to the automotive electronic cruise control system instance in AUTOSAR. The experimental results show that the proposed method has good performance in the aspects of jitter time, blocking time, frequency of scheduling and data traffic.

Keywords Automotive electronics, ECU configuration, Runnable entity, Task, Mapping

1 简介

近年来,随着汽车电子技术的发展以及人们对汽车舒适性和安全性需求的不断提高,汽车电子软件系统变得越来越复杂。为此,全球汽车制造商、供应商以及其他一些半导体和电子软件公司于2003年共同建立了AUTOSAR(AUTomotive Open System Architecture)组织,并制定了汽车开放系统架构[1],即AUTOSAR标准。

为了缩短汽车电子软件开发的周期,同时增强汽车电子应用程序的可扩展性和通用性,AUTOSAR采用了系统级和ECU级并行开发的模式。在系统级设计中,汽车电子应用程序由若干构件组成。每个构件实现了应用程序的某一特定功能。若干构件通过相互协作共同实现应用程序的具体行为。然而,在ECU级设计中,操作系统调度的对象却是任务[2]。因此,在进行ECU配置时,须将构件内的可运行实体映射到具体任务中,以供操作系统调度执行,这个过程被称为"可运行实体-任务映射"。一种较为简单的映射方法是将每个可运行实体单独封装为一个任务,简称方法1。但是,如此大量的

到稿日期:2017-02-17 返修日期:2017-06-11 本文受国家自然科学基金(61175061/F030506)资助。
冉 正(1987-),男,博士生,主要研究方向为嵌入式系统,E-mail:ranzheng517@sina.com;罗 蕾(1967-),女,硕士,教授,主要研究方向为嵌入式系统,E-mail:liuo@uestc.edu.cn(通信作者);晏 华(1970-),女,博士,副教授,主要研究方向为嵌入式软件、计算智能;李 允(1971-),男,博士,教授,研究员,主要研究方向为普适计算、实时和嵌入式操作系统及其应用、嵌入式应用设计方法。

任务会使得系统被频繁调度,增加系统的调度开销。另一种较为实用的方法是将拥有相同周期的可运行实体映射为一个任务[3],简称方法 2。然而,该方法忽略了可运行实体之间的依赖关系和数据通信,且缺乏在映射过程中对任务实时性的分析。在实际中,可运行实体-任务映射主要根据 AUTOSAR 标准中关于任务生成的规范[4],依靠软件工程师的丰富经验来手动完成。目前的自动映射方法(如方法 1 和方法 2)还不能很好地满足汽车电子系统对实时性的需求。

因此,本文针对可运行实体到任务的映射过程中任务对实时性的需求,提出自动映射方法,并将其应用于 AUTOSAR 标准的汽车电子巡航控制系统实例中。实验结果证实了本方法的有效性。

2　任务模型

2.1　构件与可运行实体

在 AUTOSAR 架构中,汽车电子应用软件由构件(Software Component)组成[5]。构件的具体行为依靠可运行实体以及可运行实体之间的相互协作来实现。通常情况下,一个构件包含一个或多个可运行实体(Runnable Entity)。可运行实体是一段用于实现一个简单算法或某一特定功能的程序代码。在此,定义构件为:$C=\{c_i \mid i=1,2,\cdots,N\}$,$c_i=\{r_j \mid j=1,2,\cdots,n_i\}$,其中,$C$ 为一个 ECU 中的构件集合,c_i 和 r_j 分别为构件和构件中的可运行实体,n_i 为构件 c_i 中可运行实体的数量。

可运行实体之间的通信一般分为构件内部的通信和构件之间的通信。位于同一构件的可运行实体可以共享构件内部的变量,而位于不同构件之间的可运行实体则只能通过端口通信。一般而言,每个构件存在若干个端口(Port),这些端口或用于数据通信,或用于功能调用。用于发送数据或提供服务的端口被称为 P-Port,而用于接收数据或请求服务的端口被称为 R-Port。

在此,可运行实体之间的通信关系可定义为 $N(C,TR,SD,DC)$,$TR=\{tr(r_p,r_q)\}$,$SD=\{sd(r_p,r_q)\}$,$DC=\{dc(r_p,r_q)\}$,其中,C 为构件集合,$tr(r_p,r_q)$ 表示 r_p 会触发 r_q,$sd(r_p,r_q)$表示 r_p 和 r_q 之间的共享数据,$dc(r_p,r_q)$表示可运行实体 r_p 到 r_q 的通信数据,如图 1 所示。

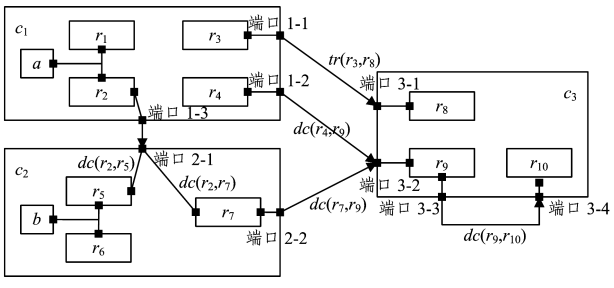

图 1　构件之间的通信关系

Fig. 1　Communication relationship between componenets

在图 1 中,不同构件的可运行实体可以通过端口相互通信;而同一构件中的可运行实体既可以通过端口通信,也可以通过构件内部数据共享的方式交互。例如,图 1 中 a 和 b 分别为构件 c_1 和 c_2 的内部变量。r_1 和 r_2 共享数据 a,r_5 和 r_6 共享数据 b。$dc(r_2,r_5)$,$dc(r_2,r_7)$,$dc(r_4,r_9)$,$dc(r_7,r_9)$则表示不同构件之间的可运行实体可以通过端口进行通信,而 $dc(r_9,r_{10})$表示位于同一构件内的可运行实体之间也可以通过端口进行通信。$tr(r_3,r_8)$则是功能调用,即 r_3 触发 r_8 运行。

2.2　RTE-事件

RTE-事件(RTE-Event)在 AUTOSAR 系统级设计中用于触发可运行实体的执行[5],AUTOSAR 大致定义了 12 种不同类型的 RTE-事件[5]。这些 RTE-事件大多是周期性的,即每隔一个固定的时间段触发一次。严格的周期性 RTE-事件由底层计时器直接触发[6],而另一些周期性 RTE-事件则由周期性可运行实体产生,即一个周期性可运行实体在执行的过程中可能会产生一系列 RTE-事件[7]。这些 RTE-事件虽然也呈现出周期性,但由于任务抢占、任务阻塞等原因而有着明显的抖动。因此,由这些 RTE-事件所触发的可运行实体也存在着明显的抖动,即抖动会从以上一个可运行实体传递到下一个运行实体[8]。在此,定义可运行实体为 $r_j=(P(r_j),e(r_j),J(r_j))$。其中,$P(r_j)$为可运行实体的周期,$e(r_j)$为可运行实体的最坏执行时间,$J(r_j)$为可运行实体的抖动时间。对于具有严格周期性的可运行实体而言,$J(r_j)=0$。

例如,假设 r_1 是严格周期性可运行实体,r_2 则受 r_1 的触发而周期性地执行。但是,由于 r_1 在运行的过程中可能会被其他高优先级的任务抢占或因共享资源而被阻塞,因此,r_2 和 r_3 存在明显的抖动,如图 2 所示。

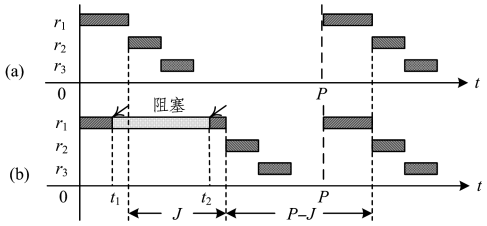

图 2　可运行实体激活抖动

Fig. 2　Jitter of runnable entities

在图 2 中,r_1,r_2 和 r_3 的周期均为 P。r_1 在执行结束时触发 r_2,r_2 在执行结束时触发 r_3。在图 2(a)中,r_1 在执行过程中未受到阻塞。因此,r_2 也呈现出严格的周期性。然而,在图 2(b)中,r_1 在$[t_1,t_2]$时间内被阻塞,这导致 r_2 的激活时间延迟了 J 个单位。如果 r_2 还要激活其他可运行实体(如 r_3),那么它们的激活时间也将延迟。因此,r_2 的激活时间不仅存在着明显的抖动,而且会扩散到其他直接或间接受 r_2 激活的可运行实体中。

2.3　任务

作为 AUTOSAR 操作系统调度的基本单位,任务为可运行实体提供了诸如上下文和堆栈空间等常用资源。因此,一个或多个关联性较强的可运行实体往往被映射到一个任务中。在此,定义任务为 $\Gamma=\{\tau_i \mid i=1,2,\cdots,K\}$,$\tau_i=\{r_j \mid j=1,2,\cdots,k_i\}$,其中 k_i 为任务 τ_i 中可运行实体的数量。定义任务的时间属性为 $\tau_i=(Pri(\tau_i),P(\tau_i),e(\tau_i),HP(\tau_i))$,其中,$Pri(\tau_i)$为任务的优先级,$P(\tau_i)$为任务的周期,$e(\tau_i)$为任务的最坏执行时间,$HP(\tau_i)$为任务的长周期(hyper-period)。

所有可运行实体在任务中顺序执行,且具有一个激活偏移值(Offset),即在任务开始后,需等待若干个单位时间才开始执行[9]。这里定义映射到任务之后的可运行实体的属性为 $\tau_i(r_j)=(P(r_j),e(r_j),I(r_j),O(r_j))$,其中 $P(r_j)$ 和 $e(r_j)$ 分别为可运行实体的周期和最坏执行时间,$I(r_j)$ 为 r_j 在任务 τ_i 中的执行顺序,$O(r_j)$ 为 r_j 的激活偏移值。根据 AUTOSAR 规范,任务的周期为任务中所有可运行实体的周期和激活偏移值的最大公约数,且不能小于可运行实体的最坏执行时间。任务的长周期为所有可运行实体的周期的最小公倍数。

例如,假设一个任务中有 4 个可运行实体:$\tau_1(r_1)=(50,4,1,0)$,$\tau_1(r2)=(100,4,2,0)$,$\tau_1(r_3)=(100,4,3,70)$,$\tau_1(r_4)=(50,4,4,20)$,那么任务的周期为 $GCD(50,0,100,0,100,70,50,20)=10$,任务的长周期为 $LCM(50,100,100,50)=100$。任务中可运行实体在一个长周期内的执行情况如图 3 所示。

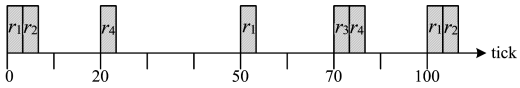

图 3 可运行实体的执行顺序

Fig. 3 Execution order of runnable entities

在图 3 中,任务每隔 10 个节拍调度一次,但任务中的可运行实体却并非在每次任务调度中都执行。例如,在任务第 1 次执行中(即[0,10]区间)只有可运行实体 r_1 和 r_2 执行,而在任务第 2 次执行中(即[10,20]区间)就没有可运行实体执行。可运行实体的最坏执行时间为 $e(r_1)+e(r_2)=e(r_3)+e(r_4)=4+4=8<10$,因此,该任务可以有效地调度执行。然而,$r_1$ 和 r_4 每执行一次,任务需要调度 5 次;而 r_2 和 r_3 每调度一次,任务需要调度 10 次。因此,将不同周期的可运行实体映射到一个任务内,将会使任务被频繁调度。

2.4 可运行实体-任务映射

在可运行实体到任务的映射过程中,可运行实体之间的依赖关系和数据通信与任务的执行情况密切相关。因此,可运行实体的映射策略将会严重影响任务的执行效率。

本文主要从以下 4 个方面来优化映射策略。

1)顺序触发

一般而言,顺序触发的可运行实体都具有相同的周期。将其映射到一个任务之后,任务的周期不变。在任务每一次调度中都会将其中的可运行实体顺序执行一遍,而非单独调度每个可运行实体,从而减少了系统调度的频率。

另外,对于顺序触发的可运行实体,将其映射到一个任务中可以消除抖动。定义任务的抖动时间为:

$$J(\tau_i)=\sum_{\forall r_p\in\tau_i,r_q\notin\tau_i,tr(r_p,r_q)} J(r_p) \tag{1}$$

目标 1:对于所有任务,最小化任务的抖动时间,即 $\forall\tau_i$,Min $J(\tau_i)$。

2)数据一致性

可运行实体之间不仅存在着 RTE-事件触发关系,而且存在着数据共享,如图 1 中的 a 与 b。因此 AUTOSAR 规范了诸如中断屏蔽(Interrupt Blocking Strategy)、信号量锁(Semaphore Locks)等机制来保证可运行实体之间数据的一致性[5]。但是这些机制可能会造成任务阻塞(Task Blocking)[10]。任务之间的共享数据量越大,读写共享数据的时间越长,造成的加锁时间也就越长,任务的可能阻塞时间也就越长。但是,如果将两个存在共享变量的可运行实体映射到一个任务中,那么这个共享变量就成为了任务的内部变量。在任务内部,可运行实体顺序执行,从而避免了阻塞。定义共享数据 $sd(r_p,r_q)$ 的加锁时间为 $LT(r_p,r_q)$,那么任务的阻塞时间为:

$$B(\tau_i)=\sum_{\forall r_p\in\tau_i,r_q\notin\tau_i,\exists sd(r_p,r_q)} LT(r_p,r_q) \tag{2}$$

目标 2:对于所有任务,最小化任务的阻塞时间,即 $\forall\tau_i$,Min$B(\tau_i)$。

3)系统调度频繁度

对于那些具有不同的周期但又有较强数据关联性的可运行实体,如果将其映射到一个任务中,那么任务的周期等于任务中所有可运行实体的周期的最大公约数,如图 3 所示。这样,任务的周期变短,任务被调度得更加频繁,从而使得系统的调度开销增大。

定义一个长周期内所有任务调度的总次数作为系统调度的频繁度,即:

$$F=lcm(P(r_1),\cdots,P(r_N))\times\left(\frac{1}{P(r_1)}+\cdots+\frac{1}{P(r_K)}\right) \tag{3}$$

系统的调度频繁度仅作为系统中所有任务的总体评估。然而对于单个任务而言,如果其周期过短,甚至小于任务中可运行实体的执行时间,任务将不可调度。因此,在目标 3 中,对所有任务的周期均做最大化处理,以避免某个任务因周期过短而被频繁调度。

目标 3:对于所有任务,最大化任务的周期,即 $\forall\tau_i$,Max $P(\tau_i)$。

4)系统的总体通信开销最小化

AUTOSAR 将可运行实体之间的通信分为任务内部通信、分区内部通信、分区之间通信和 ECU 之间通信。对于底层系统而言,不同区域之间的通信开销是不同的[5]。本文主要考虑单个 ECU 内部的可运行实体到任务的映射,因此,将可运行实体之间的通信分为任务内部的通信和任务之间的通信。在 AUTOSAR 架构中,任务内部的可运行实体可通过任务的内部变量通信,而任务间的通信不仅需要额外的数据缓冲区,而且通信机制也更加复杂。因此,任务的内部通信开销小于任务之间的通信开销[11]。如果可运行实体之间的数据通信在任务内部进行,那么系统的通信开销将会减少。定义端口通信 $dc(r_p,r_q)$ 的数据量为 $DS(r_p,r_q)$,那么任务间的数据通信量为:

$$Cost=\sum_{\forall r_p\in\tau,r_q\notin\tau,\exists dc(r_p,r_q)} DS(r_p,r_q) \tag{4}$$

目标 4:最小化任务间数据的通信量,即 Min $Cost$。

3 自动映射方法

本文主要通过 3 步来完成可运行实体到任务的自动映射。首先,处理可运行实体之间的触发关系,存在触发关系且具有相同周期的可运行实体首先被映射到一个任务;然后尽可能将存在共享数据的任务进行合并;最后,根据之前的任务映射情况,合并存在大量通信数据的任务。

3.1 顺序触发

在开始映射之前,首先为每个可运行实体创建一个单独

任务,即每个任务中只有一个可运行实体;然后,以每个任务为顶点,以 RTE-事件中的触发关系为边,建立有向图。例如,根据图 3 中的 10 个可运行实体 r_1-r_{10} 分别创建 10 个任务 $\tau_1-\tau_{10}$,如图 4 所示。

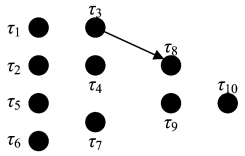

图 4　触发关系图

Fig. 4　Dependency graph of triggering

由于在图 3 中只存在一个触发关系,即 τ_3 触发 τ_8,因此图 4 中只有一条从 τ_3 到 τ_8 的有向边。

接下来,采用深度优先遍历找出触发关系图的所有连通子图,然后合并每个连通子图中的任务。这样,那些具有触发关系的任务将会被合并到一个任务中,从而消除触发调用带来的任务抖动。例如,图 4 中的任务 τ_3 为严格周期性任务且不存在抖动($J(\tau_3)=0$),但任务 τ_8 由于受任务 τ_3 的触发而运行,因此存在明显抖动($J(\tau_8)\neq 0$)。因此,将任务 τ_3 与任务 τ_8 合并为一个任务 τ_3'。合并之后,新任务 τ_3' 明显消除了原来存在于任务 τ_8 中的抖动($J(\tau_3')=0$)。

假设可运行实体的总数为 N。根据可运行实体之间的触发关系图,建立触发关系表 $TR[N][N]$。

$$TR[i][j]=\begin{cases}1, & \tau_i \text{ 触发 } \tau_j\\ 0, & \tau_i \text{ 不触发 } \tau_j\end{cases} \tag{5}$$

TSK 为任务中的可运行实体集合。算法的伪代码如算法 1 所示。

算法 1　TR_TASK

输入:触发关系表 $TR[N][N]$

输出:任务集合 TSK

1. 找出所有的严格周期性可运行实体 r_1-r_n,并建立 n 个初始任务 TSK{1}-TSK{n}。
2. for i←1 to n
3.　TR_DEEP_TASK(i);
4. end

TR_DEEP_TASK(t_i)

输入:任务号 t_i

输出:任务 TSK{t_i}

1. for i←1 t on
2.　if TR[t_i][i]=1
3.　　TSK{t_i}←TSK{t_i}∪TSK{i};
4.　　TR_DEEP_TASK(i);
5.　end
6. end

算法 1 遍历了触发关系图中的所有严格周期性可运行实体和所有的触发关系。由于一个可运行实体要么受一个严格周期性 RTE-事件触发,要么受其他某个可运行实体触发,因此在搜索过程中,一个可运行实体仅被搜索一次。本文中,RTE-事件的数量等于严格周期性可运行实体的数量与触发关系的数量之和,即 n,因此算法 1 的复杂度为 $O(n)$。

3.2　共享数据

合并存在数据共享的两个任务虽然可以消除因加锁而造成的任务阻塞,但是这两个任务可能具有不同的周期,合并之后的周期将会变小,甚至小于任务的执行时间。这显然与目标 3 相悖。因此,在处理共享数据时,不得不兼顾任务周期。首先建立任务间的数据共享表 $SD[n][n]$和周期加权表 $WT[n][n]$,n 为经过算法 1 处理后的任务数。数据共享表 SD 用于存储共享数据的加锁时间,即 $SD[i][j]=LT(\tau_i,\tau_j)$。周期加权表 WT 中的每项表示两个任务合并之后的新任务的周期。如果新任务的周期小于新任务的执行时间,则表示这两个任务不能合并,将表中新任务的周期赋值为 0。

$$WT[i][j]=\begin{cases}GCD(P(\tau_i),P(\tau_j)), & GCD(P(\tau_i),P(\tau_j))>e(\tau_i)+e(\tau_j)\\ 0, & GCD(P(\tau_i),P(\tau_j))<e(\tau_i)+e(\tau_j)\end{cases} \tag{6}$$

周期加权表主要用于过滤和加权数据共享表。数据共享表 SD 与周期加权表 WT 之积为任务组合判定表 DT,即 $DT[i][j]=SD[i][j]\times WT[i][j]$。$DT$ 用于判定两个任务是否合并为一个任务,如图 5 所示。

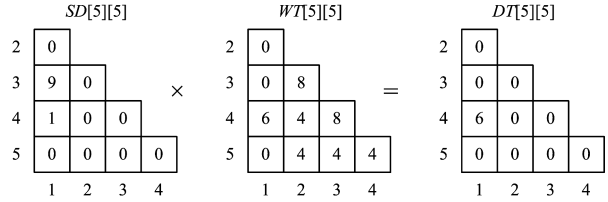

图 5　任务组合判定表

Fig. 5　Task combination decision table

图 5 中存在着两个共享数据,分别是 $SD[3][1]=9$ 和 $SD[4][1]=1$,它们所需的加锁时间分别为 9 和 1。但是,因为 $WT[3][1]=0$,所以任务 τ_1 与任务 τ_3 是不能合并的。对于任务 τ_1 与任务 τ_4,如果它们合并为一个任务,那么合并后的任务周期为 6,将其乘以共享数据所需的加锁时间 1,即得到在任务合并判定表中的数值 6。

接下来,采用层次聚类(Hierarchical Clustering)的方法[12]合并那些具有共享数据的任务。在聚类之前,首先根据算法 1 中任务组合的结果计算任务的周期 tP 和任务的最坏执行时间 te,并更新共享数据表 SD,同时计算初始的周期过滤表 WT。然后,在每次聚类的过程中,寻找 DT 表中的最大值 $DT[t_i][t_j]$。最后,合并任务 τ_{ti} 和 τ_{tj},并更新 tP,te,SD,WT 和 DT。算法的伪代码如算法 2 所示。

算法 2　SD_TASK

输入:TSK,SD,P,e

输出:TSK

1. 根据算法 1 的结果,初始化 tP,te,SD,WT;
2. DT←SD×WT;
3. 找出 DT 表中的最大值 DT [t_i][t_j];
4. while DT [t_i][t_j]>0
5.　TSK{t_i}←TSK{t_i}∪TSK{t_j};
6.　t_n←t_{n-1};更新 tP,te,SD,WT;
7.　DT←SD×WT;

8. 找出 DT 表中的最大值 DT [t_i][t_j];

9. end

算法 2 中,输入参数 TSK 为算法 1 的结果,SD 为共享数据表,P 和 e 分别为可运行实体的周期和最坏执行时间。由于在每次迭代过程中都需要处理空间复杂度为 $O(n\times n)$的图表,因此每一次迭代的算法的复杂度为 $O(n^2)$。算法 2 的时间复杂度为 $O(m\times n^2)$,其中 m 为迭代次数,且 $m<n$。

3.3 数据通信

对数据通信的处理方法与 3.2 节类似。首先建立任务间的数据通信表 $DC[n'][n']$和周期过滤表 $FT[n'][n']$,其中 n'为经过算法 2 处理之后的任务数。数据通信表 DC 中的每一项表示两个任务之间的数据通信量,即 $DC[i][j]=dc(\tau_i,\tau_j)$。对于周期过滤表 FT,如果两个任务 τ_i 和 τ_j 合并之后的任务周期等于其中一个任务的周期,那么 $FT[i][j]$为 1,否则为 0。

$$FT[i][j]=\begin{cases}1, & GCD(P(\tau_i),P(\tau_j))=P(\tau_i)\vee GCD(P(\tau_i),P(\tau_j))=P(\tau_j)\\ 0, & GCD(P(\tau_i),P(\tau_j))\neq P(\tau_i)\wedge GCD(P(\tau_i),P(\tau_j))\neq P(\tau_j)\end{cases}\tag{7}$$

与 3.2 节中的周期加权表不同,周期过滤表不会改变任务的周期,也不会增加任务调度的频繁度。周期过滤表只是将那些存在数据通信且周期相同或周期成倍数的任务进行合并。同理,经过周期过滤表 FT 过滤之后的数据通信表就成为了新的任务组合判定表 DT',即 $DT'[i][j]=DC[i][j]\times FT[i][j]$。

本节同样采用层次聚类的方法合并那些存在数据通信的任务,具体方法与 3.2 节类似,不再赘述。

4 实验

本文抽取了 AUTOSAR 规范中关于汽车电子巡航控制部分的实例并加以完善。然后,依次通过算法 1、算法 2 和算法 3 得出可运行实体到任务的映射结果。最后,从任务集的抖动时间、阻塞时间、调度频繁度和任务间的通信开销 4 个方便比较了本方法(简称方法 3)与方法 1(每个可运行实体单独成为一个任务)、方法 2(将周期相同的可运行实体映射到一个任务中)的结果。

实验环境为 Windows7 64-bit 操作系统,Intel(R) Core (TM) i5-4460 CPU@3.2 GHz 以及 8 GB 内存。编译环境为 MATLAB version 8.3.0.532 64-bit。

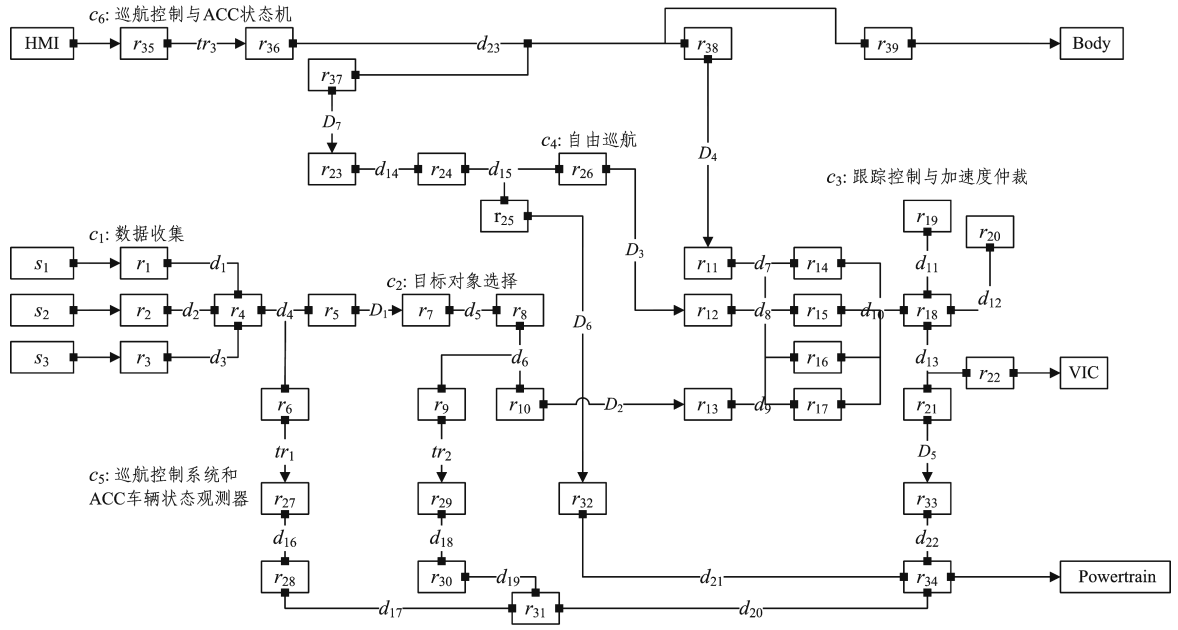

图 6 汽车电子巡航控制系统

Fig. 6 Automotive adaptive cruise control system

4.1 实验数据

汽车电子巡航控制系统[13]主要由 6 个构件组成,分别为传感器数据收集、目标对象选择、自由巡航控制、跟踪控制与加速仲裁、巡航控制与 ACC 状态机以及巡航控制和 ACC 车辆观测器。本文根据这 6 个构件的功能和通信拓扑结构构建了 39 个可运行实体、3 个触发关系、7 个构件间的数据通信和 23 个构件内的共享数据,如图 6 所示。其中,39 个可运行实体(r_1-r_{39})的周期与执行时间如表 1 所列。7 个构件间的数据通信量(D_1-D_7)如表 2 所列。23 个共享数据(d_1-d_{23})的加锁时间均为 1 μs。

表 1 可运行实体的周期与执行时间

Table 1 Period and execution time of runnable entities

(单位:μs)

r	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9
P	20	20	20	60	30	30	30	60	30
e	1	1	1	1	1	1	1	2	1
r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}	r_{17}	r_{18}	r_{19}
30	120	120	120	240	240	240	240	120	40
1	1	1	1	2	2	2	2	2	1
r_{20}	r_{21}	r_{22}	r_{23}	r_{24}	r_{25}	r_{26}	r_{27}	r_{28}	r_{29}
40	120	120	60	120	120	120	30	60	30
1	1	1	1	2	1	1	1	1	1
r_{30}	r_{31}	r_{32}	r_{33}	r_{34}	r_{35}	r_{36}	r_{37}	r_{38}	r_{39}
60	120	60	120	240	30	30	60	60	120
1	2	1	1	2	1	2	1	1	1

表 2　构件间的数据通信量

Table 2　Communication data amount between components

(单位:Bytes)

D_1	D_2	D_3	D_4	D_5	D_6	D_7
40	20	10	30	10	10	30

4.2　实验结果

在经过本方法中的算法 1、算法 2、算法 3 之后,巡航控制程序中的 39 个可运行实体被映射到 17 个任务中,分别为 $\tau_1=\{r_1\},\tau_2=\{r_2\},\tau_3=\{r_3,r_5,r_7,r_8\},\tau_4=\{r_6,r_{27},r_{28}\},\tau_5=\{r_9,r_{29},r_{30},r_{31}\},\tau_6=\{r_{10},r_{13},r_{14}\},\tau_7=\{r_{15}\},\tau_8=\{r_{16}\},\tau_9=\{r_{17},r_{18}\},\tau_{10}=\{r_{19}\},\tau_{11}=\{r_{20}\},\tau_{12}=\{r_{21},r_{33},r_{34}\},\tau_{13}=\{r_{22}\},\tau_{14}=\{r_{26},r_{12}\},\tau_{15}=\{r_{35},r_{36}\},\tau_{16}=\{r_{37},r_{38},r_{11},r_{23},r_{24},r_{25},r_{32}\},\tau_{17}=\{r_{39}\}$。

1)抖动时间

图 6 中存在 3 个触发关系,分别为 $tr(r_6,r_{27})$,$tr(r_9,r_{29})$,$tr(r_{35},r_{36})$。因此,r_{27},r_{29},r_{36}的抖动时间分别为 r_6,r_9,r_{35}的阻塞时间与抢占时间之和。因此,在方法 1 中有 3 个任务存在抖动。由于存在触发关系的任务具有相同的周期,而在方法 2 中,周期相同的可运行实体都被映射到了一个任务中,因此任务集中不存在有抖动的任务。在方法 3 中,那些存在触发关系的可运行实体也均被映射到了相同的任务中,因此所有任务均不存在抖动。

2)阻塞时间

图 7 中的(a)—(c)分别表示在方法 1、方法 2、方法 3 中共享数据加锁对每个任务造成的阻塞时间。在方法 1、方法 2、方法 3 中,所有可运行实体的总加锁时间分别为 30 μs,26 μs 和 6 μs。从图 7 中的比较可以看出,在经过方法 3 之后,共享数据加锁对任务造成的阻塞时间大大减少。

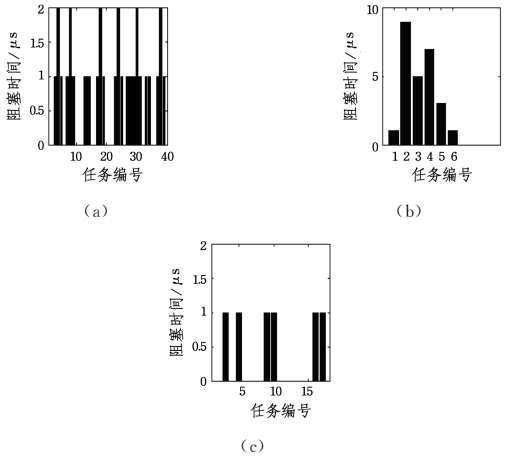

图 7　共享数据加锁时间

Fig. 7　Lock time of shared data

3)调度频繁度

从表 1 中可以看出,任务的长周期(可运行实体周期的最小公倍数)为 240μs。在方法 1、方法 2、方法 3 中,所有任务在一个长周期内分别被调度了 181 次、33 次和 108 次。虽然方法 2 的调度频繁度最低(这是因为所有周期相同的可运行实体都映射到了一个任务),但是方法 3 在兼顾了可运行实体之间的触发关系、数据共享关系以及数据通信关系之后,依然降低了系统的调度频繁度,如图 8 所示。

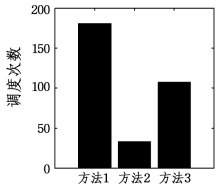

图 8　调度频繁度

Fig. 8　Scheduling frequency

(4)数据通信量

在方法 1、方法 2、方法 3 中,任务之间的数据通信量分别为 150 Bytes,60 Bytes 和 0 Bytes。方法 3 将任务之间的数据通信量降到了最低值,如图 9 所示。

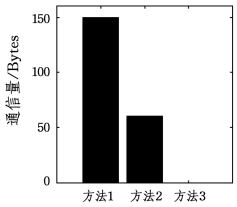

图 9　数据通信量

Fig. 9　Communication data amount

结束语　本文根据汽车电子 ECU 配置的实际需求,提出了可运行实体到任务的自动映射方法。该方法结合了深度优先遍历与层次聚类算法,并将周期加权表与周期过滤表引入层次聚类算法中,消除了层次聚类算法中将距离作为单一的聚类判定标准的局限性。将本方法用于汽车电子巡航控制系统中,并将其与以前的方法进行比较,结果显示,本方法减少了任务的抖动和阻塞时间,降低了系统的调度频繁度,减少了任务间的通信开销。

参 考 文 献

[1] FÜRST S,BECHTER M. AUTOSAR for Connected and Autonomous Vehicles:The AUTOSAR Adaptive Platform [C]// 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop. 2016:215-217.

[2] PARTNERSHIP A. Specification of Operating System (V4. 1. 0R4. 0 Rev 2)[OL]. http://www. autosar. org. 2010.

[3] ZHANG M,GU Z H. Optimization Issues in Mapping AUTOSAR Components To Distributed Multithreaded Implementations[C]// 22nd IEEE International Symposium on Rapid System Prototyping(RSP). 2011:23-29.

[4] LONG R S,LI H,PENG W,et al. An Approach to Optimize Intra-ECU Communication Based on Mapping of AUTOSAR Runnable Entities [C] // International Conference on Embedded Software and Systems. 2009:138-143.

[5] PARTNERSHIP A. Specification of RTEV(3. 1. 0R4. 0 Rev 2) [OL]. http://www. autosar. org. 2010.

[6] HU M L,LUO J,WANG Y,et al. Scheduling periodic task graphs for safety-critical time-triggered avionic systems [J]. IEEE Transactions on Aerospace and Electronic Systems,2015,51(3):2294-2304.

[7] XIE G Q,ZENG G,LI Z T,et al. Adaptive Dynamic Scheduling on Multi-functional Mixed-Criticality Automotive Cyber-Physical Systems [J]. IEEE Transactions on Vehicular Technology,2017(99):1-15.

(下转第 226 页)

[12] WANG Y,LEI Y J. A technique for constructing intuitionistic fuzzy entropy[J]. Control and Decision,2007,22(12):1390-1394.(in Chinese)
王毅,雷英杰. 一种直觉模糊熵的构造方法[J]. 控制与决策,2007,22(12):1390-1394.

[13] VERMA R,SHARMA B D. Exponential entropy on intuitionistic fuzzy sets[J]. Kybernetika,2013,49(1):114-127.

[14] YE J. Two effective measures of intuitionistic fuzzy entropy[J]. Computing,2010,87(1):55-62.

[15] WEI C P,GAO Z H,GUO T T. An intuitionistic fuzzy entropy measure based on trigonometric function[J]. Control and Decision,2012,27(4):571-574.(in Chinese)
魏翠萍,高志海,郭婷婷. 一个基于三角函数的直觉模糊熵公式[J]. 控制与决策,2012,27(4):571-574.

[16] WANG J Q,WANG P. Intuitionistic linguistic fuzzy multi-criteria decision-making method based on intuitionistic fuzzy entropy[J]. Control and Decision,2012,27(11):1694-1698.(in Chinese)
王坚强,王佩. 基于直觉模糊熵的直觉语言多准则决策方法[J]. 控制与决策,2012,27(11):1694-1698.

[17] GAO M M,SUN T,ZHU J J. Revised axiomatic definition and structural formula of intuitionistic fuzzy[J]. Control and Decision,2014,29(3):470-474.(in Chinese)
高明美,孙涛,朱建军. 一种改进的直觉模糊熵公理化定义和构造公式[J]. 控制与决策,2014,29(3):470-474.

[18] LIU M F,REN H P. A study of multi-attribute decision making based on a new intuitionistic fuzzy entropy measure[J]. System Engineering Theory and Practice,2015,35(11):2909-2916.(in Chinese)
刘满凤,任海平. 基于一类新的直觉模糊熵的多属性决策方法研究[J]. 系统工程理论与实践,2015,35(11):2909-2916.

[19] ZHAO F,WANG Q S,HAO W L. Improvement of constraint conditions and new constructional method for intuitionistic fuzzy entropy[J]. Journal of Computer Applications,2015,35(12):3461-3464.(in Chinese)
赵飞,王青山,郝万亮. 直觉模糊熵约束条件的改进及新模糊熵构造[J]. 计算机应用,2015,35(12):3461-3464.

[20] GAO W,LIU S,HUANG L. Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique[J]. Communications in Nonlinear Science and Numerical Simulation,2012,17(11):4316-4327.

[21] ZHOU Y,BAO L,CHEN C L P. A new 1D chaotic system for image encryption[J]. Signal Processing,2014,97(7):172-182.

[22] THONGMOOL G,PHANKOKKRUAD M. Analysis of interaction user interface patterns and usability study in computer assisted instruction for Tablet PC[C]// 4th IEEE International Conference on Control System,Computing and Engineering. Batu Ferringhi:IEEE Press,2014:472-477.

[23] WETCHAKORN T,PROMPOON N. Method for mobile user interface design patterns creation for iOS platform[C]// 12th International Joint Conference on Computer Science and Software Engineering. Songkhla:IEEE Press,2015:150-155.

[24] BIRTOLO C,RONCA D. Advances in clustering collaborative filtering by means of fuzzy c-means and trust[J]. Expert Systems with Applications,2013,40(17):6997-7009.

[25] PAUL A,MAITY S. Kernel fuzzy c-means clustering on energy detection based cooperative spectrum sensing[J]. Digital Communications and Networks,2016,2(4):196-205.

[26] WU Z,XIE W,YU J. Fuzzy c-means clustering algorithm based on kernel method[C]//5th International Conference on Computational Intelligence and Multimedia Applications. Xi'an:IEEE Press,1995:1942-1948.

[27] NEIL T. Mobile Design pattern gallery:UI patterns for mobile applications[M]. Sebastopol:O'Reilly Media,2012.

[28] HUBERT L,ARABIE P. Comparing partitions [J]. Journal of Classification,1985,2(1):193-218.

[29] DAVIES L D,BOULDIN W D. A cluster separation Measure [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,1979,1(2):224-227.

[30] CHEN W,SONG Y,BAI H,et al. Parallel spectral clustering in distributed systems[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence,2011,33(3):568-586.

[31] STREHL A,GHOSH J. Cluster ensembles-Knowledge reuse framework for combining multiple partitions[J]. Journal of Machine Learning Research,2003,3(3):583-617.

(上接第 195 页)

[8] KAI R. Compositional Scheduling AnalysisUsing Standard Event Models [D]. Braunschweig:Technical University Carolo-Wilhelmina of Braunschweig,2005.

[9] MONOT A,NAVET N,BAVOUX B,et al. Multisource Software on Multicore Automotive ECUs-Combining Runnable Sequencing With Task [J]. Scheduling IEEE Transactions on Industrial Electronics,2012,59(10):3934-3942.

[10] FERRARI A,NATALE M D,GENTILE G,et al. Time and memory tradeoffs in the implementationof AUTOSAR components [C]// Conference on Design,Automation and Test in Europe. 2009:864-869.

[11] FARAGARDI H R,LISPER B,SANDSTRÖM K,et al. A Communication-Aware Solution Framework for MappingAUTOSAR Runnables on Multi-core Systems [C]// Proceedings of the 2014 IEEE Emerging Technology and Factory Automation. 2014:1-9.

[12] HAN J W,KAMBER M,PEI J. Data Mining:Concepts and Techniques (Third Edition) [M]. Burlington: Morgan Kaufmann. 2011:456-461.

[13] PARTNERSHIP A. Explanation of Application Interfaces of the Body and Comfort Domain (V1. 2. 0 R4. 0 Rev 2) [OL]. http://www. autosar. org,2010.