小 型 微 型 计 算 机 系 统
Journal of Chinese Computer Systems
DOI:10.20009/j.cnki.21-1106/TP.2025-0159
2026 年 5 月 第 5 期
Vol.47 No.5 2026

安全增强的车载嵌入式系统中任务映射和调度算法

万 果,魏叶华,易宣成,孙治杰,李江伟

(湖南师范大学 信息科学与工程学院,长沙 410000)
E-mail:guooow@hunnu.edu.cn

摘 要: 随着现代车载嵌入式系统应用的不断集成与电子控制单元(ECU)数量的增长,系统实时性保障面临更大挑战.同时,汽车与外部环境交互频繁,具有灵活数据速率的控制器局域网(CAN FD)虽提升了传输性能,但仍缺乏内置安全机制,易受伪装等攻击威胁.而添加安全机制往往会占用实时性资源,危害车辆安全,因此必须在保证系统实时性的同时提升安全性.为此,本文提出了一种基于强化学习的任务映射和调度算法(Reinforcement Learning-based Task Mapping and Scheduling Algorithm,RLMS),将任务映射建模为马尔科夫决策过程,结合资源感知机制以 ECU 利用率为约束优化方案,在满足实时性约束的前提下减少 CAN FD 总线消息数量,并为消息提供 4 字节 MAC 的基础安全保护.为进一步提升系统安全性,设计安全增强机制(Security Enhancement with Balanced Rounds,SEBR),利用系统空闲时间逐轮扩展消息的 MAC 字节.最终,通过真实案例和模拟实验验证了所提方法的有效性.

关 键 词: 嵌入式系统;CAN FD;任务调度;安全性增强
中图分类号: TP399 **文献标识码:** A **文 章 编 号:** 1000-1220(2026)05-1166-09

Task Mapping and Scheduling Algorithms for Security-enhanced Automotive Embedded Systems

WAN Guo,WEI Yehua,YI Xuancheng,SUN Zhijie,LI Jiangwei
(College of Information Science and Engineering,Hunan Normal University,Changsha 410000,China)

Abstract: As modern in-vehicle embedded systems continue to integrate more applications and the number of Electronic Control Units (ECUs) increases, ensuring system real-time performance becomes increasingly challenging. Meanwhile, frequent interactions between vehicles and the external environment raise security concerns. Although the Controller Area Network with Flexible Data-Rate (CAN FD) improves transmission performance through flexible data rates, it lacks built-in security mechanisms and remains vulnerable to threats such as spoofing. Introducing security mechanisms often consumes real-time resources and may compromise vehicle safety. Therefore, enhancing security while ensuring real-time guarantees is essential. This paper proposes a Reinforcement Learning-based Task Mapping and Scheduling Algorithm (RLMS), which formulates the task mapping process as a Markov Decision Process. By incorporating a resource-aware mechanism that constrains ECU utilization, the proposed method reduces the number of CAN FD bus messages while satisfying real-time constraints. Each message is provided with basic security protection through a 4-byte Message Authentication Code (MAC). To further strengthen system security, a mechanism called Security Enhancement with Balanced Rounds (SEBR) is introduced, which gradually increases the MAC length by leveraging system idle time in a round-by-round manner. The effectiveness of the proposed approach is validated through real-world case studies and simulation experiments.

Keywords: embedded systems; CAN FD; task scheduling; security enhancement

0 引 言

随着智能驾驶和车联网技术的不断发展,现代汽车电子系统的功能日益丰富,同时车辆内部集成的电子控制单元(ECU)数量也显著增加,这使得汽车电子系统已经演变为复杂的分布式车载嵌入式系统.这些 ECU 通过多种车载网络协议(如 CAN、LIN、FlexRay 等)和网关实现高频数据交互,以协同控制车载应用.为降低硬件成本并适应系统复杂性的提升,通常多个任务需集成到一个 ECU 上执行[1].然而,这种资源共享模式在提高硬件利用率的同时易引发实时性资源竞争,可能因计算和通信资源冲突导致部分任务无法在截止时间内执行完成,从而威胁车辆功能安全甚至引发事故.而车载嵌入式系统具有严格的实时性约束,要求系统必须在时限内执行任务,以保障驾驶安全和用户体验[2].例如,自动紧急制动系统(AEBS)需要在检测到前方障碍物后的几毫秒内自动执行刹车操作,以有效避免碰撞.这种应用集成给任务映射和调度带来了严格的实时性挑战.

CAN 是目前车载嵌入式系统中应用最广泛的网络总

收稿日期:2025-04-10 收修改稿日期:2025-05-23 基金项目:国家自然科学基金项目(62072175)资助. 作者简介:**万 果**,女,2001 年生,硕士研究生,研究方向为车载系统设计优化算法;**魏叶华**,男,1979 年生,博士,教授,CCF 高级会员,研究方向为嵌入式系统、智能网联汽车安全;**易宣成**,男,2002 年生,硕士研究生,研究方向为车联网信息安全;**孙治杰**,男,2002 年生,硕士研究生,CCF 会员,研究方向为车载网络安全通信;**李江伟**,男,2002 年生,硕士研究生,CCF 会员,研究方向为车载网络安全.

线[3],随着数据传输需求的增长,CAN 因数据带宽和帧长度限制已无法满足数据密集型应用[4]. 为此,博世公司推出了 CAN FD 标准,显著提升了帧长度和传输速率,以支持更复杂的网络通信需求[5]. 但与 CAN 类似,CAN FD 同样面临安全问题. 其缺乏内置的安全认证机制[4],易受伪装等攻击威胁. 伪装攻击是车载通信中最常见的攻击且难以检测,攻击者通过伪造消息冒充合法节点,可能导致车辆控制系统的错误决策,甚至引发严重的安全事故[6]. 尤其是汽车的网联化使得汽车与外部环境交互的接口增多,更易受到网络攻击威胁[7]. 因此,引入安全认证机制以提高车辆网络安全性势在必行,但其额外开销往往会消耗嵌入式系统实时性资源、增加端到端延迟. 因此,如何设计高效的任务到 ECU 映射与任务调度方法,并且在保证系统实时性的同时提升通信安全性,已成为车载系统早期设计阶段中的关键挑战.

针对车载系统中实时性与安全性保障的挑战,已有研究从任务映射、调度优化、通信安全机制设计等方面进行了深入探索. Yong 等人[8]提出了一种集成时序和安全约束的混合模型,使用混合整数线性规划算法(Mixed-Integer Linear Programming,MILP)将信号打包成 CAN FD 消息,并通过在消息中添加 4 字节 MAC 提供安全性. Roy S K 等人[9]为优化共享平台上共享链路的任务图调度,设计了一种基于整数线性规划的任务调度方案,在提高调度精度的同时减少任务延迟和避免冲突. 然而,MILP 等确定性方法随着任务规模的增加其计算复杂度将显著上升. Anna 等人[10]利用启发式算法优化多任务调度,以降低 ECU 上汽车应用集成对控制性能的影响,并通过优化端到端延迟以提升系统整体控制效果. Shane 等人[11]针对高级驾驶辅助系统平台,开发了结合遗传算法和模拟退火的任务调度和消息传递的优化方法,以确保满足汽车系统的时序要求. 这些方法优化任务调度,保障了系统实时性,但未考虑到消息的信息安全.

Xie 等人[12]提出了一种前后向探索技术,通过动态调整 MAC 大小应对伪装攻击同时确保 CAN FD 消息的实时性. Ma 等人[13]设计了一种两阶段 CAN-FD 消息打包方案,在安全约束下降低带宽利用率并提高信号接受率. 这些方法增强了 CAN FD 上消息集的安全性,但未考虑优化任务映射. 为兼顾安全性和实时性,Stumpf 等人[14]提出为 ECU 配备硬件安全模块(Hardware Security Module,HSM),而硬件模块的使用增加汽车的成本. Yong 等人[15]设计了一种基于干扰平衡的启发式算法(IBH),在任务映射、调度和消息调度中满足安全性和时序约束的同时降低了 HSM 的硬件数量. Niklas 等人[16]提出了基于模拟退火的元启发式算法,优化了时间敏感网络以满足安全关键应用的实时性和安全性需求,但启发式算法容易陷入局部最优.

综上,为了在保证车载嵌入式系统实时性的同时应对安全威胁,本文提出了一种基于强化学习的任务映射和调度算法. 相比于确定性方法和启发式方法的局限性,基于强化学习的优化算法能够更好地适应复杂动态的任务环境. 通过模拟“Agent”与“环境”交互,算法无需依赖精确模型即可收集反馈并持续优化策略[17],相比依赖固定假设的确定性方法更具灵活性. 其探索新策略和利用已知有效策略之间的平衡机制则使其更具全局优化潜力. 本文的主要贡献如下:

1)设计了一种基于强化学习的任务映射和调度算法(RLMS),将任务映射问题建模为马尔科夫决策过程,并利用 Q-learning 迭代优化映射策略. 在此基础上,引入基于资源感知约束的动态优化,以 ECU 资源利用率为约束生成任务映射方案,避免过载并优化资源分配. 该方案在满足系统实时约束的前提下,最小化 CAN FD 总线上的消息暴露数量,并为消息提供 4 B MAC 的基础安全措施.

2)提出一种安全增强机制(SEBR),在调度完成后利用系统空闲时间,逐轮增强消息的 MAC 字节数来提升通信安全性. 该机制以安全增益为核心指标并结合消息成本重要度构建安全增强等级,优先增强等级高的消息,从而在满足实时性约束的前提下,以尽可能少的轮次提升系统整体安全性.

3)通过真实案例和仿真实验验证了所提出算法的有效性. 实验结果表明,RLMS 算法能有效生成可行的任务映射和调度方案,在满足实时性约束的前提下将总线消息率降低至 20% 以下. 此外,所提出的安全增强算法能够在相对较少轮次下添加更多的 MAC 字节,增强了安全性.

1　系统模型

1.1　CAN-FD 网络模型

车载网络由多个 ECU 组成,ECU 通过 CAN-FD 总线相连. 在系统运行中,任务由 ECU 执行,而消息通过总线完成数据传输. 图 1 展示了一个局部车载网络模型. 其中 ECU_1 和 ECU_2 通过 CAN-FD 总线相连,传感器连接 ECU_1 而执行器连接到 ECU_2. 车载应用由传感器触发并在 ECU_1 上执行任务 t_1,ECU_1 通过 CAN-FD 总线将消息 $m_{1,2}$ 发送到 ECU_2,从而触发任务 t_2 在 ECU_2 执行. 最终通过执行器完成指令.

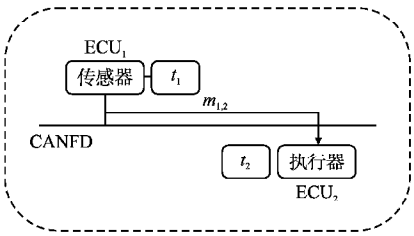

图 1　基于 CAN FD 的车载网络模型

Fig. 1　In-vehicle network model based on CAN FD

系统使用 CAN FD 协议,带宽为 12MBit/s,有效载荷高达 64B. 它由 7 个部分组成,包括帧起始 SOF,仲裁段,控制段,数据域,CRC 域,ACK 域,帧结束. SOF 标志着报文的开始;仲裁字段用于节点优先级仲裁;控制段包括数据长度码(DLC)和帧标识符,分别指示数据字段长度和帧类型;数据域用于携带有效载荷信息;CRC 域包括了一个循环冗余校验(CRC)序列和 CRC 界定符位,用于错误检测;ACK 域通过 ACK 位确认帧接收;帧结束表示一帧信息的结束. 在传输中,仲裁阶段比特率限制为 1Mbps,而数据阶段最高可达 10Mbps. 在本文设定中,数据传输阶段的比特率被设定为 8Mbps.

1.2　应用模型

车载应用(G)指车载系统中实现特定功能的软件模块,

通常由多个任务组成,本文将其建模为有向无环图(DAG),其中节点集 $T=\{t_i,i=1,2,\cdots,n\}$ 表示任务,边集 $M=\{m_{i,j},i=1,2,\cdots,n;j=1,2,\cdots,n\}$ 表示任务间消息. 任务 t_i 的直接前驱任务和后继任务分别记为 $pre(t_i)$ 和 $suc(t_i)$. 无前驱但有后继的任务为入口任务,有前驱但无后继的任务为出口任务. 任务间存在依赖关系,当前任务仅在其前驱任务完成且接收到所有消息后才能执行,任务执行完成后传输消息给后继任务. 每个应用使用 T*i* 和 D*i* 分别代表周期和截止期. 所有应用的超周期定义为 T_{cycle}. 为便于求解,引入虚拟入口任务 t_{entry} 和虚拟出口任务 t_{exit},以整合多个 DAG 为一个整体.

任务在 ECU 上所需最坏情况下执行时间(WCET)示例如表 1 所示. 由于 ECU 呈现异构特性,同一个任务在不同 ECU 上的 WCET 不同,其中"N/A"表示该任务无法映射到该 ECU.

表 1 任务在各 ECU 上的 WCET(μs)示例

Table 1 Example of task WCET(μs) on each ECU

	t_{10}	t_{11}	t_{20}	t_{21}	t_{22}	t_{23}	t_{24}	t_{25}	t_{30}	t_{31}	t_{32}	t_{33}
ECU_1	N/A	30	20	30	N/A	20	60	20	40	60	40	N/A
ECU_2	40	50	30	N/A	60	30	90	60	90	100	50	40

总线消息 $m_{i,j}$ 的最坏传输时间(WCTT)与消息中有效载荷(Payload,$p_{i,j}$)的大小有关[12]. 有效载荷是 CAN FD 数据段内传输的数据,其大小必须符合以下标准值:0、1、2、3、…、32、48、64 字节. 消息的 WCTT 计算式如式(1)[18]计算. 本文设定中数据场传输数据的比特率为 8 MBit/s,而仲裁场传输数据的比特率为 1 MBit/s,即 $t_{data}=0.125\mu s$ 和 $t_{arb}=1\mu s$.

$$WCTT(m_{i,j})=32t_{arb}+(28+5\left|\frac{p_{i,j}-16}{64}\right|+10\times p_{i,j})t_{data} \tag{1}$$

此外,假设 ECU 内部通信是安全的,且传输时间可以忽略不计. 因此,对于映射至同一 ECU 的前后继任务,消息的 WCTT 视为 0 μs. 表 2 给出了不同 Payload 大小下 CAN FD 消息的 WCTT 计算结果.

1.3 安全模型

本文考虑通过在 CAN FD 帧的数据字段尾部添加 MAC 值来增强系统防御攻击的能力. MAC 通过共享密钥和消息内容生成固定长度的认证码,供接收方验证消息的完整性和真实性,防止消息被篡改或伪造[19].

表 2 CAN FD 有效载荷对应的 WCTT

Table 2 WCTT for CAN FD payload

Payload(B)	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64
WCTT(μs)	36	38	39	40	41	43	44	45	50	55	61	66	76	96	116

在 CAN FD 总线初始化阶段,为通信的 ECU 分配共享密钥. 发送方生成 MAC 值并将其附加到消息中后传输给接收方. 接收方收到消息后,提取消息中的 MAC 值,并使用相同的密钥和算法重新计算 MAC 值进行比较. 如果两者一致,表明消息完整且来源可信;否则,认证失败. 然而,MAC 认证需要在安全性与实时性之间权衡. 通常 MAC 长度增加可降低不同消息生成相同认证码的概率,能提升抗碰撞能力,降低认证码被伪造的概率. 但也会扩展消息帧的长度,增加传输时间,进而可能影响实时性.

2 基于强化学习的任务映射和调度算法(RLMS)

2.1 目标函数

为了在满足任务在截止期内完成的实时性要求下,尽可能减少总线上的消息数量,并保证消息 4 字节 MAC 的基础安全防护,设计了如下目标函数:

$$\min(M(M)/N(M)+R(G)/D(G))\ s.t.\ R(G)\leqslant D(G) \tag{2}$$

其中 $M(M)$ 表示暴露在总线上的消息,$N(M)$ 表示为暴露的消息数和 ECU 间进行安全传输的消息数之和;$R(G)$ 表示应用的总体完成时间,$D(G)$ 表示预定的截止期限. 所有应用和任务 $D(G)$ 相同.

2.2 算法整体架构

RLMS 算法包含 3 个阶段,如图 2 所示,输入应用的 DAG 数据,在第 1 阶段将任务映射问题建模为马尔可夫决策过程,采用 Q-learning 算法迭代求解直至收敛,生成任务映射的基础Q表,以充分挖掘潜在的全局最优解. 第2阶段,结合

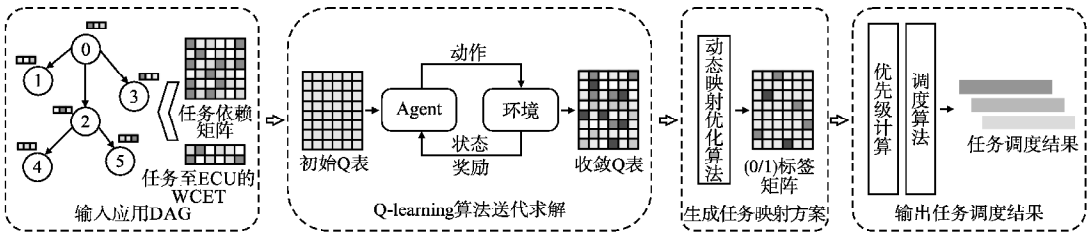

图 2 算法流程

Fig. 2 Algorithm flow

基于资源约束的动态映射优化算法,以 ECU 资源利用率为约束,将 Q 表转化为任务映射到 ECU 的(0/1)标签矩阵,生成初步任务映射方案,避免了过载问题并优化资源分配. 最后,采用异构最早完成时间算法(Heterogeneous Earliest Finish Time,HEFT)[20]对任务进行优先级排序,并通过非抢占式静态优先级调度策略生成最终调度表.

2.3 基于 Q-learning 的任务映射

Q-learning 的 3 个核心要素为 Agent、环境和奖励函数. 在设计的方法中,Agent 为执行任务映射决策的智能体,其目标是基于系统状态选择最优的任务映射策略. 环境为 Agent 交互的对象,即任务调度系统,包括车载系统模型和应用模型等. 它接收 Agent 的决策,并返回新的系统状态和奖励. 奖励

函数用于评估每次决策的效果,指导 Agent 学习. Q-learning 通过 Agent 与环境的反复交互,以试错方式逐步学习最优策略[21].

2.3.1　马尔可夫决策过程(MDP)建模

首先将任务映射问题建模为 MDP,通过定义状态空间 S、动作空间 A 和奖励函数 r,描述系统的动态行为和优化目标. 系统的状态空间 S 定义为所有待映射任务的集合,如下:

$$S=(s(\tau_1),\cdots,s(\tau_i),\cdots,s(\tau_n)) \tag{3}$$

在任意时刻 t,状态 $s(\tau_i)$ 表示当前待映射的任务 τ_i,用于指示系统中的映射进度,使系统能够依次遍历所有任务并明确制定映射决策. Agent 在当前状态 $s(\tau_i)$ 中从所有等待映射的任务中选择一个任务进行映射,执行后系统状态更新为 $s'(\tau_{i+1})$,决策过程转入对 τ_{i+1} 的映射. 随着任务逐步映射,状态空间 S 的未映射任务子集逐渐缩减,直至所有任务完成映射.

动作 a 代表 Agent 处于特定状态 s 下所选择的任务映射策略,指将等待映射的当前任务 τ_i 映射给选定的 ECU 节点. 动作空间 A 为所有可行动作的集合,其表示如下:

$$A=(a_1,\cdots,a_i,\cdots,a_m) \tag{4}$$

Agent 在当前状态 $s(\tau_i)$ 下执行任务映射动作 a 后,该任务被添加到所选 ECU 的队列中,随后系统进入下一状态 $s'(\tau_{i+1})$ 等待下一次动作选择过程. Agent 通过在特定状态下选择动作,并从环境中接收反馈来探索环境. 反馈根据奖励函数来定义,合理的奖励函数能够有效引导 Agent 学习最优策略,对强化学习的效果和性能至关重要. 为实现优化目标,本文结合目标函数设计奖励函数. 因此奖励函数的组成包含两个方面:1)时序约束的奖励;2)总线消息数量的奖励. 为确保任务在截止期内完成,定义了基于任务完成时间的奖励项,该奖励根据应用完成时间 *makespa*n 是否满足截止期要求来赋值. 具体如下:

$$rdl=\begin{cases}0, & \text{if } makespan\leqslant D(G)\\ -k\cdot(makespan(G)-D(G)), & \text{if otherwise}\end{cases} \tag{5}$$

当任务 $makespan\leqslant D(G)$ 即在预设的截止期范围内,为避免过度干预 Agent 的学习,奖励值设为 0,使其有更多可能性去探索不同的映射方案;然而,由于时序约束是硬性要求,当 *makespan* 超时,则给予较大的惩罚,且超时时间越长,惩罚越重,以促使 Agent 减少不可行的行为. 此设计旨在赋予 Agent学习自由度,助其探索最优任务映射方案,提升映射效率. 为实现总线消息数最小化,消息传递奖励机制将评估暴露在总线上的消息数量,设计如下:

$$r_{\text{msg}}=\begin{cases}\text{p}, & \text{if no new messages are exposed}\\ -\Delta mn(G), & \text{if otherwise}\end{cases} \tag{6}$$

其中 p 为一个常数,当没有消息暴露时,给予正向奖励 p,鼓励 Agent 选择减少增加消息的映射方案;$\Delta mn(G)$ 为增加的消息数量,当新增消息时,则给予惩罚,惩罚幅度与消息数成正比. 该设计旨在尽可能减少 ECU 之间通信,从而降低通信负载和潜在的被攻击风险. 总奖励函数 r 由 r_{dl} 和 r_{msg} 组成,以平衡任务调度的实时性与通信效率:

$$r=r_{\text{dl}}+r_{\text{msg}} \tag{7}$$

当 *makespan* 满足截止期时,r_{dl} 为 0,鼓励探索低通信开销的映射方案;若超时,则给予超时惩罚,使其优先满足实时性约束. 同时,r_{msg} 通过奖励和惩罚引导减少总线消息. 这一设计旨在通过联合优化时序和通信,在确保任务按期完成的同时减少总线负载,提高系统资源利用效率与调度可靠性.

2.3.2　任务映射

Q-learning 的核心是通过构建 Q 表 $Q(s,a)$,存储在每个状态 s 下执行动作 a 获得的长期累积奖励的期望值,并不断更新 Q 值来引导 Agent 接近最优解. 本文将任务作为 Q 表的行,ECU 作为列,Q 值表示任务映射到 ECU 的期望奖励值. 初始时,Q 值设为固定值,Agent 依据奖励 r 和预估的未来奖励更新 Q 值,更新规则遵循 Bellman 方程[22],其公式如下:

$$Q(s,a)=(1-\alpha)\cdot Q(s,a)+\alpha\cdot(r+\gamma\cdot\max_{a'}Q(s',a')) \tag{8}$$

其中 $Q(s,a)$ 为当前状态-动作对的 Q 值. α 为学习率即 $0\leqslant\alpha\leqslant1$,控制每次更新对 Q 值的影响程度. r 是 Agent 在状态 s 采取动作 a 后获得的即时奖励. γ 为折扣因子,即 $0\leqslant\gamma\leqslant1$,用于考虑未来累积奖励的重要性. $\max_{a'}Q(s',a')$ 表示在下一个状态 s' 中选择最优动作 a' 所对应的 Q 值.

Agent 的动作选择策略对学习效果至关重要,需在“探索”新方案与“利用”已有经验之间找到平衡. 若缺乏足够的探索,算法可能会过早收敛;而过于依赖探索则可能导致学习效率低下. 因此,本文采用 ε-greedy 策略指导动作选择. Agent 以概率 ε 随机选择一个动作,即搜索新方案;而以概率 $1-\varepsilon$ 选择当前已知的最优动作,即具有最大 Q 值的动作,动作选择遵循以下规则:

$$\pi(a/s)^*=\begin{cases}\arg\max_a Q(s,a) & \text{with probability } 1-\varepsilon\\ \text{random action} & \text{with probability } \varepsilon\end{cases} \tag{9}$$

在训练初期,设置较大 ε 值以扩大探索空间,使 Agent 增加状态-动作对的信息,从而发现潜在的全局最优解. 随着训练进行,逐步减小 ε 值,使 Agent 聚焦于利用已有经验,加速收敛.

Q-learning 任务映射过程如算法 1 所示. 将学习率 α、折扣因子 γ、ε、应用和 ECU 的执行时间表作为输入,初始化 Q 表. 在每个 episode 中,Agent 通过选择和评估映射决策,更新 Q 表,以反映当前决策表现. 每个 episode 独立运行完整的任务映射周期,环境在 episode 开始时重置,但 Q 表中的数据会保留并用于指导后续决策,以累积学习经验. 第 4 ~ 10 行中对于每个状态 $s(\tau_i)\in S$,Agent 依据 ε-greedy 策略选择动作. 若随机数 y 小于 ε,则从可选节点中随机选择一个进行映射,否则采取具有最高 Q 值的动作. 随后,第 11 ~ 13 行根据状态-动作对计算即时奖励 r、系统转移至新状态 s',最后更新 Q 值. 算法重复执行 episode,直到达到预设的终止条件即最大时间步数.

算法 1. 基于 Q-learning 的任务映射算法

输入: $\alpha,\gamma,\varepsilon$,DAG(G),WCET(G)

输出: A Q-table π^*

1. 初始化:$Q(s,a)\rightarrow0,\forall s\in S,a\in A$
2. **for** each *episode* **do**
3.　　初始化 S、A and *Environment*
4.　**for** each $s(\tau_i)\in S$ **do**
5.　　$y\rightarrow$Uniform(0,1)
6.　　**if** $y<\varepsilon$

7. $a \rightarrow$RandomAction(A)
8. **else**
9. $a \rightarrow \max Q(s,a')$
10. 执行动作 a(i.e.,分配任务 τ_i至 ECU_a)
11. $r \rightarrow$ Reward(s,a)//计算即时奖励
12. $s \rightarrow s'$
13. Q(s,a)→ Q(s,a) + α [r + γ max Q(s′,a′) − Q(s,a)]//更新 Q 表
14. **end**
15. **end**

2.3.3 动态映射优化

收敛后的 Q 表的 Q 值反映了任务映射在不同 ECU 上的预期累积奖励. Q 值越大,表明任务更倾向于被映射到对应的 ECU. 为获得具体的映射方案,将 Q 表转换为任务映射到 ECU 的(0/1)标签矩阵. 然而,直接依据最大 Q 值进行任务映射可能导致两个问题:一是任务集中映射到某些 ECU,可能造成局部资源饱和,进而调度结果超出截止时间,造成调度失败;二是任务映射不均,部分 ECU 负载过重,而其他 ECU 未被充分利用,降低了整体资源利用率. 为解决上述问题,设计了一种基于资源约束的动态映射优化算法,在映射任务时实时评估 ECU 的资源占用情况. 每个任务 Q 表中的 Q 值按降序排序,依次尝试将其映射到 Q 值最大的 ECU. 在映射过程中动态检查映射到同一 ECU 的任务的 WCET 总和是否超出该 ECU 最大资源利用率:79%[23]. 若超出限制,则放弃映射当前 ECU,选择次优 Q 值对应的 ECU 重新映射任务. 在每个任务确定映射到合适的 Q 值对应的 ECU 后,将 Q 值对应的位置标记为 1,表示该任务已映射至该 ECU,而其他未选中的位置则标记为 0,最终形成任务映射到 ECU 的(0/1)标签矩阵. 通过这种资源感知的调整策略,以缓解 ECU 负载不均和任务过载风险,提升资源分配的合理性与调度的全局优化效果.

2.4 任务调度

在获得任务映射标签矩阵后,通过 HEFT 算法计算任务的优先级. HEFT 算法依据任务的依赖关系计算优先级,保证合理的调度顺序,提升整体调度效率. 任务 t_i 的优先级计算如式(10)所示:

$$priority(t_i) = \mathrm{WCET}(t_i) + \frac{\sum_{t_j \in suc(t_i)} I_{i,j} \cdot \mathrm{WCTT}(m_{i,j})}{\sum_{t_j \in suc(t_i)} I_{i,j}} + \max_{t_j \in suc(t_i)} priority(t_j) \tag{10}$$

其中 $suc(t_i)$是任务 t_i 后继任务的集合;$I_{i,j}$是指示变量,当 t_i 和 t_j 映射到不同的 ECU 上时 $I_{i,j}=1$,否则 $I_{i,j}=0$. 在计算任务优先级时综合考虑了 3 项因素:任务的计算开销越大,优先级越高;其次是任务间通信开销较大时优先级提高,保障关键通信优先调度;最后是后继任务优先级越高,当前任务的优先级也相应提高. 基于式(10)计算各任务的优先级后,按照优先级值由高到低进行排序,从而确定任务的静态优先级顺序.

在确定任务映射方案和优先级后,本文采用非抢占式静态优先级调度方法对任务进行调度,生成最终的调度方案. 调度过程中,按照任务优先级依次调度各任务,对于每个任务,依据依赖关系在其映射的 ECU 上选择最早可用的时间窗口,并确保所有前驱任务完成且满足通信延迟约束. 调度遵循非抢占式约束,即任务一旦开始执行不可中断,同时保证同一 ECU 上各任务时间片无重叠.

3 安全增强机制(SEBR)

3.1 增强目标

任务调度完成后,若系统仍有空闲时间,将逐轮增加 MAC 字节数,以进一步提升消息的安全性. 但若只考虑 MAC 增加量而忽略轮次带来的安全时间成本,将导致额外的资源开销. 因此,本文将平衡 MAC 增加量和轮次,旨在满足实时性约束前提下通过较少的轮次提升系统安全性. 其目标函数如下:

$$\max \frac{\sum_{m_{i,j} \in M} \mathrm{MAC}(m_{i,j})}{\sum_{m_{i,j} \in M} Round(m_{i,j})} \quad s.t. \quad R(G) \leqslant D(G) \tag{11}$$

其中 MAC($m_{i,j}$)表示消息 $m_{i,j}$ 中的 MAC 字节数,$Round$($m_{i,j}$)表示消息 $m_{i,j}$ 进行安全增强的轮数. 在安全增强过程中,系统的响应时间 $R(G)$必须在截止时间 $D(G)$内.

3.2 安全增强等级

定义 1. 系统空闲时间(SIT):应用完成时间和截止期之间的差为系统空闲时间. 由于 MAC 字节数的增加会导致消息的 WCTT 增加,从而增加应用响应时间. 为保障系统实时性约束,所有消息 WCTT 增加总和需限制在 SIT 范围内. 其计算如下:

$$SIT(G) = D(G) - R(G) \tag{12}$$

定义 2. 消息延迟容限(mdl):消息传输可以延迟的最晚时间,定义为消息目标任务的开始时间与 CAN FD 总线上紧接其后的下一条消息开始时间之间的最小值. 其计算如式(13)所示,其中 $st(t_j)$为消息目标任务的开始时间,$sm(m'_{i,j})$为后继消息的开始时间.

$$mdl(m_{i,j}) = \min(st(t_j), sm(m'_{i,j})) \tag{13}$$

定义 3. 消息松弛时间(Δmdt):消息的实际传输完成时间与其消息 mdl 之间的差距,表示在不增加应用整体完成时间的情况下,消息传输所允许的额外时间分配. 其计算如式(14)所示,其中 $fm(m_{i,j})$为消息传输完成时间.

$$\Delta mdt(m_{i,j}) = mdl(m_{i,j}) - fm(m_{i,j}) \tag{14}$$

安全增益(SG)衡量安全增强所带来的安全性提升. 由于系统资源有限,无法对所有消息同时提供最高级别的安全保护,因此优先对 SG 更高的消息进行优化,以在有限资源下提升整体安全性. SG 定义为消息安全增强后 MAC 字节增加量与完成时间增加量之比,其计算如下:

$$SG(m_{i,j}) = \begin{cases} \Delta mac(m_{i,j}), \text{if } \Delta mdt(m_{i,j}) \geqslant \Delta wctt(m_{i,j}) \\ \dfrac{\Delta mac(m_{i,j})}{\Delta wctt(m_{i,j}) - \Delta mdt(m_{i,j})}, \text{if otherwise} \end{cases} \tag{15}$$

$\Delta mac(m_{i,j}) = \mathrm{MAC}et - \mathrm{MAC}ct$, $\Delta wctt(m_{i,j}) = \mathrm{WCTT}et - \mathrm{WCTT}ct$. $\Delta mac(m_{i,j})$和 $\Delta wctt(m_{i,j})$分别表示 MAC 字节数和消息传输时间的变化,前者反映安全提升,后者衡量时间成本. MACet 和 WCTTet 为进行安全增强后的消息的 MAC 值和 WCTT 值,而 MACct 和 WCTTct 为增强前的值. 在消息传输时间增加仍在 Δmdt 范围内时,只需关注 MAC 字节的增加,此时 MAC 的增加不会延长应用总体完成时间,应尽可能添加 MAC 字节,以提高安全性. 但当应用的总体完成时间增加时,除了增加的 MAC 字节外,还需考虑实时性约束.

优先选择 SG 最高的消息进行安全增强,有助于添加更多 MAC 值,但忽略了轮次时间成本,而成本与增强轮次成正比.进一步观察发现,消息 Payload 的大小显著影响安全增强的效率.由于 CAN FD 的 Payload 长度需满足特定的合法大小,消息 Payload 越小,达到目标 MAC 字节数所需增强轮次越多;而 Payload 越大,所需轮次则越少.为此,定义"消息成本重要度(MCI)"值,考虑消息 Payload 对增强轮次的影响,以优化 SG 等级.表 3 展示了 MCI 与消息的 Payload 之间的关系.

表 3　消息成本重要度

Table 3　Message cost importance

Payload(B)	1	2	3	4	5	6	7	8	12	16	20	24	32	48	64
MCI	1	2	3	4	5	6	7	8	9	10	11	12	13	14	–

从表 3 中可以看出,Payload 越接近 64B,其对应的 MCI 值越高,说明其增强效率越高.例如,当需要增加到 20B 的 MAC 值时,Payload 为 12B 的消息比 Payload 为 2B 的消息能更少次数增强到 20B.因此 Payload 越大,视其重要度越高.最终的安全增强等级(SE)结合了 SG 与 MCI,用以平衡安全性能和轮次.其公式如下:

$$SE(m_{i,j})=\begin{cases} SG(m_{i,j})+MCI(m_{i,j}), & \text{if } \Delta mdt(m_{i,j})\geqslant\Delta wctt(m_{i,j}) \\ SG(m_{i,j}), & \text{if otherwise} \end{cases} \tag{16}$$

在满足条件 $\Delta mdt(m_{i,j})\geqslant\Delta wctt(m_{i,j})$ 时,将 MCI 与 SG 结合,旨在通过较少轮次添加 MAC 位以增强安全性.当安全增强导致应用总体完成时间增加时,则需要平衡安全性能与消息认证时间,以满足系统的时序约束.SE 越高表示该消息在当前轮次下的增强性价比越高.

3.3　安全增强算法

实施安全增强的过程如算法 2 所示,在每一轮迭代中,首先计算各消息增强至下一有效载荷所对应的安全增益(SG)和成本重要性(MCI),以确定其安全增强等级(SE);随后依据 SE 等级对消息进行降序排列;优先选择 SE 等级对消息进行降序排列;优先选择 SE 最高的消息进行 MAC 字节扩展,使其提升至下一有效载荷.消息增强终止条件为所有消息增强到最大载荷或者系统空闲时间耗尽.

4　实验评估

为评估所提算法的性能,本文在真实和模拟案例中进行了实验.实验平台为配备 Intel(R) Core(TM) i5 处理器(2.11GHz)、8GB RAM 及 64 位 Windows 11 操作系统的笔记本计算机,开发工具为 PyCharm,实验程序基于 Python 3.8 版本实现.对于 RLMS 算法,通过检查任务与 ECU 的调度结果,使用调度成功率(SSR)和消息率(MR)为评估指标,并与 IBH[15] 和 HGA[24] 进行了对比.SSR 表示所有任务成功调度到 ECU 上的概率,MR 为总线消息数与总消息数之比.在 SEBR 算法中,评估 MAC 字节增加数和增强轮次,并与 RSMW[18]、RSM[25] 和 RSW[25] 3 种算法进行了比较.

算法 2. SEBR 安全增强机制算法

输入: M, $D(G)$, $R(G)$

输出: MAC_{all}:MAC 增加总量;$Round_{all}$:总轮次

1. $MAC_{all}\leftarrow 0$, $Round_{all}\leftarrow 0$;
2. **while** $R(G)\leqslant D(G)$ and $SIT(G)>0$ **do**
3.　　**for** each 消息 $m_{i,j}$ *in* M **do**
4.　　计算
5.　　$\Delta mdt(m_{i,j})$, $\Delta mac(m_{i,j})$, $\Delta wctt(m_{i,j})$, $SE(m_{i,j})$;
6.　　选择具有最高安全增强等级的消息;
7.　　**if** 满足实时性约束 **then**
8.　　$MAC_{all}+=\Delta mac(m_{i,j})$, $Round_{all}+=1$;
9.　　**else**
10. continute;

4.1　真实案例实验

图 3 展示了一个真实车载系统应用案例—自适应巡航控制应用[18],该应用由 5 个 ECU 组成,并执行 32 个任务.图中节点为任务,节点值为任务在 ECU 上的平均 WCET,边为任务间的数据依赖关系,边值表示任务间传输的消息的大小.

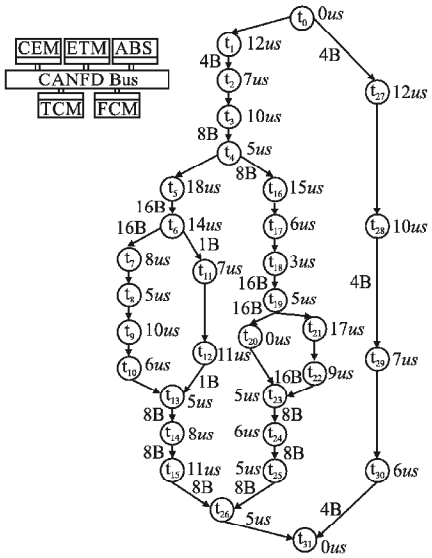

图 3　车载自适应巡航控制应用

Fig. 3　In-vehicle adaptive cruise control application

表 4　SSR 和 MR 真实案例实验结果

Table 4　SSR and MR real case experimental results

Deadline	SSR(%)			MR(%)		
	RLMS	IBH	HGA	RLMS	IBH	HGA
100	0	0	0	N	N	N
200	68	0	0	10.26	N	N
300	100	0	0	5.27	N	N
400	100	26	40	0	52.63	49.84
500	100	88	94	0	41.05	40.23
600	100	100	100	0	42.63	43.42
700	100	100	100	0	45.26	44.26
800	100	100	100	0	35.26	37.63
900	100	100	100	0	39.50	38.06
1000	100	100	100	0	42.15	40.14

表 4 展示了在不同 Deadline 下,RLMS、IBH 和 HGA 这 3 种算法的SSR和MR的实验结果,其中"N"表示未找到可行调度方案.可以看出,在SSR指标上,随着Deadline的延长,

3 种算法均能实现 100% 的 SSR. 然而,RLMS 在 200 μs 即可找到可行调度方案,并在 300 μs 实现 100% 的 SSR,而 IBH 和 HGA 需要更长的 Deadline 才能达到,同时 RLMS 的 SSR 增长速度也优于 IBH 和 HGA. 在 MR 方面,RLMS 的 MR 始终低于 IBH 和 HGA,且一直维持在较低水平. 并且,随着 Deadline 的延长,RLMS 的 MR 呈下降趋势并逐渐降至为 0,而 IBH 和 HGA 的 MR 呈现波动状态并始终保持较高的 MR. 主要原因是 RLMS 算法优先将存在消息通信的任务映射到同一 ECU,从而有效减少总线通信需求.

在真实案例中,基于映射后的固定消息集对 SEBR 算法的安全增强性能进行了验证. 结果如表 5 所示,SEBR 算法在 MAC 字节增加和轮次之间实现了良好的平衡. 随着 Deadline 的延长,所有算法添加的 MAC 字节量均有所增加. 虽然 SEBR 算法总体上 MAC 字节的增加幅度相对较小,如 Deadline 从 800 延长至 1000 时,SEBR 的 MAC 增加 168B,而 RSMW、RSM 和 RSW 分别添加了 160、176 和 188B. 但 SEBR 算法在相同 Deadline 下添加的 MAC 字节量普遍高于其他算法. 例如,在 Deadline 为 1000 时,SEBR 算法添加的 MAC,分别比其他算法多 12%、21% 和 4%. SEBR 算法添加的 MAC 字节量较高,但其所需的安全增强轮次普遍更少.

4.2 仿真实验

为进一步验证提出的 RLMS 算法的有效性和一般性,本文进行了仿真实验. 通过随机任务图生成器(RTGG)生成模拟应用,任务数范围为 24 ~ 104,以 16 为步长,ECU 数范围为 2 ~ 18,以 2 为步长. 每个任务的平均 WCET 设置为 10μs,消息的大小在 1 ~ 16 字节之间取值.

表 5 真实案例安全增强实验结果

Table 5 Real case safety enhancement experiment results

Deadline	*SLT*	Total MAC Bytes				Rounds			
		SEBR	RSMW	RSM	RSW	SEBR	RSMW	RSM	RSW
800	45	84	65	32	54	4	16	4	27
850	95	128	105	80	122	8	20	9	44
900	145	156	153	128	166	11	25	14	55
950	195	204	193	168	202	17	28	20	64
1000	245	252	225	208	242	22	32	24	69

图 4 显示了 3 种算法在 SSR 指标上的实验结果,颜色越浅表示 SSR 越高. 从图中可以看出,在任务数固定情况下随着 Deadline 的延长,3 种算法的 SSR 均显著提升. 而 RLMS 算法整体表现出更高的 SSR,尤其在任务数量和 ECU 数量较多的情况下,其 SSR 优势更为明显. 在任务数为 24 ~ 56 时,3 种算法的 SSR 差异较小,RLMS 略优于 HGA 和 IBH. 而且,随着 ECU 数量增加,IBH 和 HGA 算法的 SSR 呈下降趋势,而 RLMS 保持稳定. 当任务数超过 72 时,RLMS 的优势进一步凸显. HGA 和 IBH 的 SSR 随任务量的增加急剧下降,且随着 ECU 的增加趋近于零,而 RLMS 仍能维持较高的 SSR.

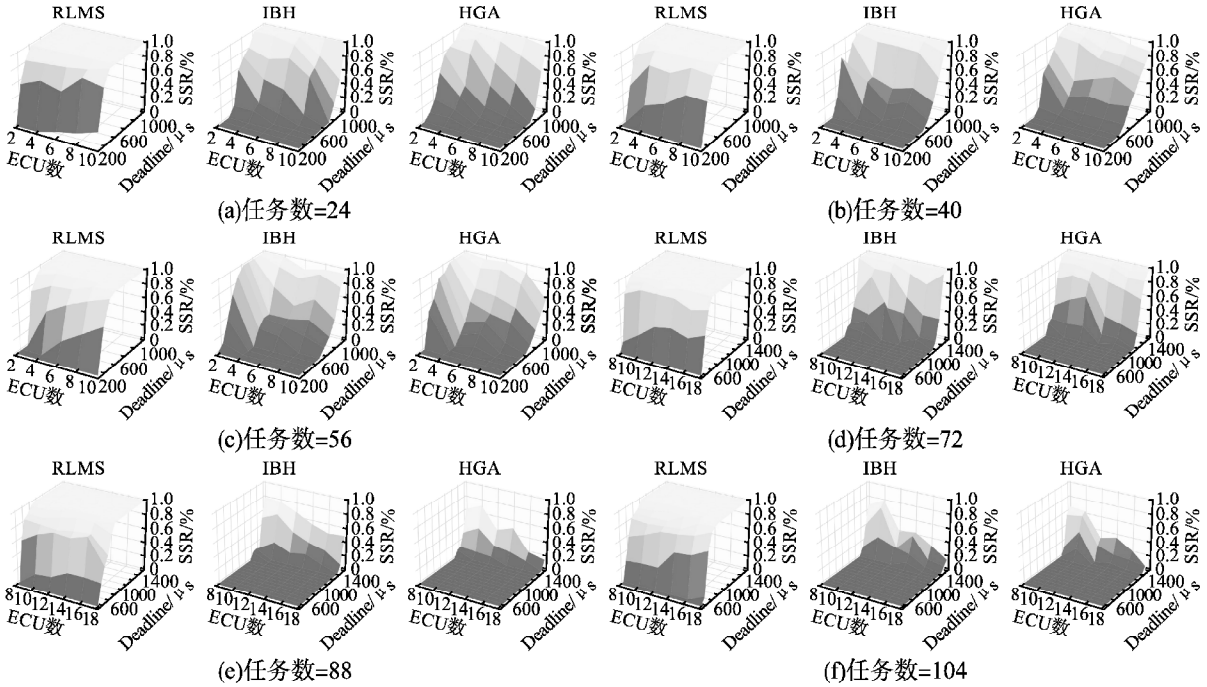

图 4 SSR 仿真实验结果

Fig. 4 SSR simulation results

图 5 展示了 3 种算法在不同 ECU 数量下的 MR 表现. 结果表明,RLMS 算法在 MR 方面显著优于 IBH 和 HGA,整体呈现出更低的 MR. 当任务数固定且 ECU 数量较少时,IBH 和 HGA 能够保持较低的 MR,但随着 ECU 数量的增加,两者的 MR 显著上升并超过 50%,而 RLMS 的 MR 基本不受影响,始终保持在 20% 以内. 且随着任务数量的增加,IBH 和 HGA 的 MR 整体上升,RLMS 则维持稳定. 结合图 4 和图 5 的结果可知,在多任务和多 ECU 的场景下,RLMS 具有显著优势. 这一优势主要源于 RLMS 算法倾向于减少消息传输,同时消息数量的减少有利于缩短应用完成时间.

为评估 SEBR 算法的性能,运用 4 种算法来增强 CAN FD 总线上消息的安全性,结果如图 6 所示. 在相同任务数量下,SEBR 算法能够普遍嵌入更多 MAC 字节数,且随着任务数量的增加,其 MAC 字节数呈显著上升趋势. 同时,SEBR 算法所需轮次相对其他算法更少,而且随着任务数的增多,其轮次增长幅度也相对较低. 基于之前的分析,该结果进一步验证了RLMS算法在复杂系统中具有良好的扩展性. 随着任务和ECU数量的增加,RLMS算法仍然能够找到优化的任务映射和调度方案,显著减少系统响应时间. 响应时间的降低为系统提供了更多松弛时间,从而能够嵌入更多 MAC 位,在提升安全性的同时满足实时性要求.

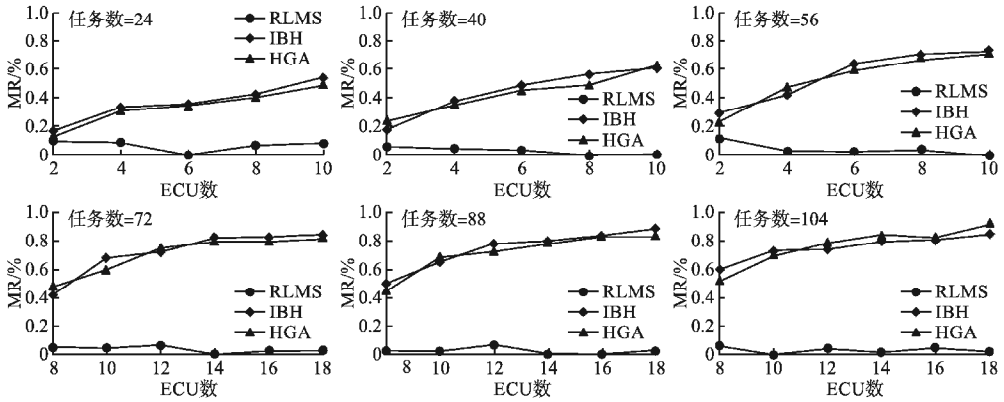

图 5　MR 仿真实验结果

Fig. 5　MR simulation results

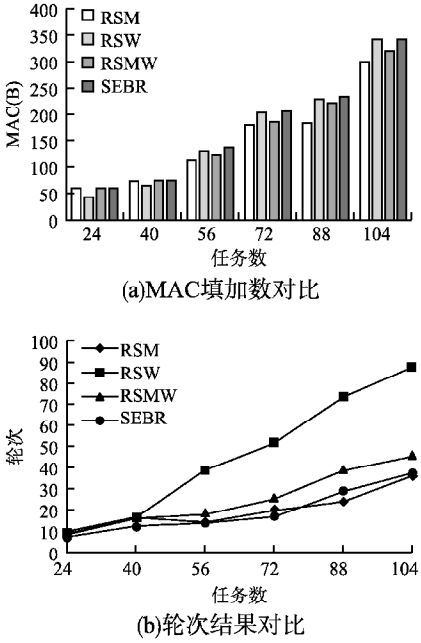

图 6　安全增强仿真实验结果

Fig. 6　Simulation results of security enhancement mechanism

5　结束语

现代车载嵌入式系统应用不断集成,ECU 和外部接口数量快速增长,显著加剧了系统在实时性和安全性方面的挑战. 为此,本文所提出的 RLMS 算法与 SEBR 机制相结合,为优化任务到 ECU 的映射和调度以及在严格的时序约束下增强消息安全性提供了全面的解决方案. RLMS 算法在保证系统实时性的前提下显著降低了总线消息暴露率,并为消息添加 4B MAC 作为基础安全措施. SEBR 算法则充分利用空余时间,通过为消息附加 MAC 字节进一步提升了安全性. 最后,通过实验验证了所提出的方法的有效性. 未来的工作拟进一步优化强化学习模型以适用更大规模的汽车应用.

References:

[1] Zou A, Li J, Gill C D, et al. RTGPU: real-time GPU scheduling of hard deadline parallel tasks with fine-grain utilization [J]. IEEE Transactions on Parallel and Distributed Systems, 2023, 34(5): 1450-1465.

[2] Sonko S, Daudu C D, Osasona F, et al. The evolution of embedded systems in automotive industry: a global review[J]. World Journal of Advanced Research and Reviews, 2024, 21(2): 96-104.

[3] Sun H, Huang W, Weng J, et al. CCID-CAN: cross-chain intrusion detection on CAN bus for autonomous vehicles[J]. IEEE Internet of Things Journal, 2024, 11(15): 26146-26159.

[4] de Andrade R, Santos M M D, Justo J F, et al. Security architecture for automotive communication networks with CAN FD[J]. Computers & Security, 2023, 129: 103203, doi: 10.1016/j.cose.2023.103203.

[5] Lodge N, Tambe N, Saqib F. Addressing vulnerabilities in CAN-FD: an exploration and security enhancement approach[J]. Internet of Things, 2024, 5(2): 290-310.

[6] Shang C, Cao J, Liu J, et al. CEAMP: a cross-domain entity authentication and message protection framework for intra-vehicle network [J]. IEEE Transactions on Intelligent Transportation Systems, 2023, 25(7): 6780-6795.

[7] Zhao R, Luo C, Gao F, et al. Application-layer anomaly detection leveraging time-series physical semantics in can-fd vehicle networks [J]. Electronics, 2024, 13(2): 377, doi: 10.3390/electronics13020377.

[8] Xie Y, Zeng G, Kurachi R, et al. Security/timing-aware design space exploration of CAN FD for automotive cyber-physical systems[J]. IEEE Transactions on Industrial Informatics, 2019, 15(2): 1094-1104.

[9] Roy S K, Devaraj R, Sarkar A. Contention cognizant scheduling of task graphs on shared bus-based heterogeneous platforms[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 41(2): 281-293.

[10] Minaeva A, Roy D, Akesson B, et al. Control performance optimization for application integration on automotive architectures[J]. IEEE Transactions on Computers, 2020, 70(7): 1059-1073.

[11] McLean S D, Juul Hansen E A, Pop P, et al. Configuring ADAS platforms for automotive applications using metaheuristics[J]. Frontiers in Robotics and AI, 2022, 8: 762227, doi: 10.3389/frobt.2021.762227.

[12] Xie G, Yang L T, Liu Y, et al. Security enhancement for real-time independent in-vehicle CAN-FD messages in vehicular networks [J]. IEEE Transactions on Vehicular Technology, 2021, 70(6): 5244-5253.

[13] Ma W, Liu Y, Xie G, et al. Security-aware CAN-FD message packing in intelligent automotive cyber-physical systems[J]. IEEE Internet of Things Journal, 2021, 9(22): 22343-22356.

[14] Stumpf F, Pohl C, Hoettges D, et al. Introducing HSM-based secure on-board communication in vehicles[C]//Proceedings of Escar Europe, 2019: 104-107.

[15] Xie Y, Guo Y, Yang S, et al. Security-related hardware cost optimization for CAN FD-based automotive cyber-physical systems[J]. Sensors, 2021, 21(20): 6807, doi: 10.3390/s21206807.

[16] Reusch N, Craciunas S S, Pop P. Dependability-aware routing and scheduling for time-sensitive networking[J]. IET Cyber-Physical Systems: Theory & Applications, 2022, 7(3): 124-146.

[17] Dabbaghjamanesh M, Moeini A, Kavousi Fard A. Reinforcement learning-based load forecasting of electric vehicle charging station using Q-learning technique[J]. IEEE Transactions on Industrial Informatics, 2021, 17(6): 4229-4237.

[18] Xie G, Yang L T, Wu W, et al. Security enhancement for real-time parallel in-vehicle applications by CAN FD message authentication [J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(8): 5038-5049.

[19] LIU Y C. Research on multi-sensor data fusion optimization algorithm in the internet of things environment[J]. Technology of IoT&AI, 2025, 57(2): 33-36.

[20] Topcuoglu H, Hariri S, Wu M Y. Performance-effective and low-complexity task scheduling for heterogeneous computing[J]. IEEE Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274.

[21] LI Y H, LIU P P, WANG W M, et al. Deep reinforcement learning task offloading method in edge end collaboration scenario[J]. Journal of Chinese Computer Systems, 2025, 46(2): 280-288.

[22] Dearden R, Friedman N, Russell S. Bayesian Q-learning[J]. Association for the Advancement of Artificial Intelligence, 1998: 761-768, doi: 10.1007/BF00993412.

[23] Buttazzo G C. Hard real-time computing systems: predictable scheduling algorithms and applications (3rd ed.)[M]. Berlin: Springer, 2011.

[24] Chauhan N K, Tyagi I, Kumar H, et al. Tasks scheduling through hybrid genetic algorithm in real-time system on heterogeneous environment[J]. SN Computer Science, 2022, 3(1): 75, doi: 10.1007/s42979-021-00959-0.

[25] Xie G, Zeng G, Kurachi R, et al. Exact WCRT analysis for message-processing tasks on gateway-integrated in-vehicle CAN clusters[J]. ACM Transactions on Embedded Computing Systems, 2018, 17(6): 1-29.

附中文参考文献：

[19] 刘裕灿. 物联网环境下多传感器数据融合优化算法研究[J]. 智能物联技术, 2025, 57(2): 33-36.

[21] 李英豪, 刘盼盼, 王文猛, 等. 边端协同场景下的深度强化学习任务卸载方法[J]. 小型微型计算机系统, 2025, 46(2): 280-288.