本发明公开了一种面向多核平台的汽车操作系统任务的高效实时调度方法，包括：在每条递归分支的起始阶段测试判断当前调度方案是否满足终止条件；对不满足终止条件的递归分支的调度方案进行剪枝判断；将当前时间节点时已完成运行的任务所属任务链的后继任务加入待调度任务列表；对待调度任务列表中的每个任务生成所有可能的调度方案，存入搜索列表，并划分递归搜索空间进行后续搜索；创建并行线程执行搜索列表中的每个调度方案，更新映射函数中的映射，并计算下一个时间节点；基于更新后的映射函数和下一时间节点，各线程并行创建新递归分支，开始递归搜索。本发明还公开了实现上述调度方法的调度系统，具有广泛应用价值。

1. 一种面向多核平台的汽车操作系统任务的高效实时调度方法，其特征在于，所述调度方法通过并行剪枝调度，寻找最优任务调度方案；包括如下步骤：

步骤一、在每条递归分支的起始阶段测试判断当前调度方案是否满足终止条件；

步骤二、对不满足终止条件的递归分支的调度方案进行剪枝判断；

步骤三、将当前时间节点时已完成运行的任务所属任务链的后继任务加入待调度任务列表；

步骤四、对待调度任务列表中的每个任务生成所有可能的调度方案，存入搜索列表，并划分递归搜索空间进行后续搜索；

步骤五、创建并行线程执行搜索列表中的每个调度方案，更新映射函数中的映射，并计算下一个时间节点；基于更新后的映射函数和下一时间节点，各线程并行创建新递归分支，开始递归搜索。

2. 如权利要求1所述的调度方法，其特征在于，步骤一中，所述终止条件是指所有任务都已存在于当前搜索到的映射函数中；

若所述递归分支当前调度方案满足所述终止条件，且优于全局映射函数ap，则在线程安全的前提下用当前搜索到的映射函数更新全局映射函数，并停止当前递归分支；

或，

若所述递归分支当前调度方案不满足所述终止条件，有任务未被调度，执行后续步骤。

3. 如权利要求1所述的调度方法，其特征在于，步骤二中，根据存在未完成任务的调度方案进行剪枝判断，包括：

调度方案的理想运行时间大于全局最优调度结果；

和/或，

调度方案违反任务链最大执行时长约束；

和/或，

调度方案违反任务起止时间约束；

当出现上述一种或多种情况时，对递归方案进行剪枝。

4. 如权利要求1所述的调度方法，其特征在于，步骤三中，根据当前时间节点T、映射函数各个任务的理论运行时间，统计当前时间节点T时已完成运行的任务；将所述已完成运行的任务所属任务链的后续任务加入待调度任务列表。

5. 如权利要求1所述的调度方法，其特征在于，步骤四中，对于待调度任务列表中每个任务，根据被分配的内核id不同，任务的优先级不同，生成所有可能的调度方案，并存入搜索列表中；根据调度方案的不同对递归搜索空间进行划分，每个调度方案将作为其所在的递归搜索空间的起始点，继续后续的搜索。

6. 如权利要求1所述的调度方法，其特征在于，步骤五中，对于搜索列表中的每个调度方案，各自分配一个线程，各个线程之间并行执行；

对于每个线程，根据所述线程中的调度方案，更新映射函数中的映射关系；根据调度方案中为任务分配的优先级关系，执行优先级最高的任务，记录各个任务的开始执行时间；根据当前时间点，以及各个运行中任务的理论运行时间，计算下一个时间节点T，要满足至少存在一个任务将在T时完成执行。

7. 一种实现如权利要求1-6之任一项所述的调度方法的调度系统，其特征在于，所述调度系统包括：DAG生成模块、任务调度模块；

所述DAG生成模块基于输入的文件信息构建DAG实例，提供任务调度信息；包括：作为调度系统核心的DAG单元、表示调度系统中任务的DAGTask单元、表示任务之间依赖关系的DAGSignal单元、表示任务执行链的TaskPath单元；

所述任务调度模块基于DAG类实例中存储的任务、任务链信息以及相关预处理数据，采用启发式搜索方法生成最优的任务调度方案；包括：记录任务执行时间的TimeNode单元、记录任务在当前启发式搜索空间中的调度状态的ResultNode单元、用于启发式搜索最佳调度方案的SimpleBacktrack单元。

8. 一种实现如权利要求1-6之任一项所述方法的硬件系统，其特征在于，所述硬件系统包括：存储器和处理器；所述存储器上存储有计算机程序，当所述计算机程序被所述处理器执行时，实现如权利要求1-6任一项所述的方法。

9. 一种计算机可读存储介质，其上存储有计算机程序，其特征在于，所述计算机程序被处理器执行时，实现如权利要求1-6任一项所述的方法。

10. 如权利要求1-6之任一项所述的调度方法，如权利要求7所述的调度系统，如权利要求8所述的硬件系统，或如权利要求9所述的计算机可读存储介质在基于汽车开放系统架构的汽车操作系统在自动驾驶场景下从图像采集、预处理、特征提取到目标识别任务调度中的应用。

**一种面向多核平台的汽车操作系统任务的高效实时调度方法、系统及应用**

**技术领域**

本发明属于嵌入式系统领域，涉及一种面向多核平台的汽车操作系统任务的高效实时调度方法、系统及应用。

**背景技术**

随着汽车智能化与电动化的加速推进，传统分布式电子电气架构的局限性日益凸显。早期车辆通过少量独立ECU（Electronic Control Unit）即可实现基础功能控制，但随着车载功能复杂度的指数级增长，现代车辆需部署数百个异构ECU。这些由不同供应商开发的ECU采用分散式硬件架构与封闭式软件系统，导致系统存在三大核心缺陷：算力资源冗余率高达30%-40%、整车线束长度超过5公里、全生命周期开发维护成本增加45%以上，严重制约了车载电子系统的可扩展性与迭代效率。为应对上述问题，虚拟化技术与汽车开放系统架构AUTOSAR（AUTomotive Open System Architecture）标准化框架的结合成为行业重要技术路径。一方面，虚拟化技术通过将物理硬件资源抽象为虚拟化层，实现多个逻辑ECU功能的集中化整合，显著减少硬件数量与线束复杂度；另一方面，AUTOSAR架构通过定义标准化的软件组件接口与任务调度机制，为复杂功能协同提供软件基础设施支持。

AUTOSAR架构包含了一个通用的软件基础设施，为定义软件组件及其接口提供了一套详尽且严谨的规范。在该架构下，系统的功能架构由一系列通过接口紧密协作的软件组件构成，这些组件共同协作以实现复杂的功能。每个软件组件的行为通过一组可运行实体（Runnable）来体现，这些可运行实体通过被映射到相应的任务来得以执行。这些任务调度机制的优化设计直接影响系统实时性能。然而，在多核硬件架构下，如何高效地实现任务调度仍是AUTOSAR系统面临的核心挑战。

当前主流的调度方案呈现两极化特征：分区调度采用静态绑核策略，任务被预先静态分配给特定的处理器，每个处理器独立地对其所分配的任务进行调度管理。这种调度方法虽能确保时序确定性，但多核处理器利用率普遍低于65%；全局调度虽可实现理论利用率上限，但其动态负载均衡机制产生的上下文切换开销，显著增加了系统能耗与延时抖动。特别是面向自动驾驶L3+级系统，其任务链端到端延时要求已压缩至100ms级，现有调度方案难以满足严格的时间确定性要求。更严峻的挑战源于系统设计复杂度的非线性增长，典型自动驾驶系统的可运行实体数量已突破5000个，任务间依赖关系形成超过10^4量级的约束条件矩阵。然而当前行业缺乏有效的设计自动化工具，导致大量的厂商仍采用人工启发式调度策略，其方案优化效率与系统规模呈现指数级负相关。这些手动设计方案往往难以满足自动驾驶领域对实时性能的极高要求。在自动驾驶这一对安全性和效率极为敏感的领域，即使运行时间的轻微过量也可能显著影响整体系统的效率，并且有可能引发重大事故，造成不可估量的损失。因此，如何在AUTOSAR架构下高效地实现任务调度方案的优化，已成为当前亟待解决的关键技术难题，对于推动嵌入式虚拟化系统在智能汽车领域落地有着重要意义。

**发明内容**

为了解决现有技术存在的不足，本发明的目的是提供一种面向多核平台的汽车操作系统任务的高效实时调度方法，通过并行剪枝调度，能够在短时间内找到最优任务调度方案。本发明正式定义了多核平台中实时系统模型和具有资源约束的任务最优调度问题，为算法设计提供了明确的数学基础；提出了一种面向多核平台的汽车操作系统任务的高效实时调度方法，通过模拟任务链的执行过程，递归搜索所有可能的调度方案，并利用并行策略和剪枝策略大幅减少搜索时间和计算复杂度；

实现本发明目的的具体技术方案是：

步骤一、终止条件设定

对于每条递归分支，在其起始阶段测试当前调度方案是否满足终止条件，所述终止条件即所有任务都已存在于当前搜索到的映射函数中。若满足，且优于全局映射函数ap，则在线程安全的前提下用当前搜索到的映射函数更新全局映射函数，并停止当前递归分支；若不满足，说明存在尚未调度的任务，继续后续步骤。

步骤二、中间结果剪枝

若递归搜索未满足终止条件，则根据当前已完成的部分任务的调度方案进行剪枝判断：计算理想运行时间，若按照当前调度方法，即使尚未调度的任务不考虑任何资源约束，其理想运行时间仍然没有优于全局最优调度结果，则进行剪枝；和/或，若当前调度方案违反任务链最大执行时长约束，则进行剪枝；和/或，若当前调度方案违反任务起止时间约束，则进行剪枝。若进行剪枝，则立刻停止当前递归分支；若不进行剪枝，则继续后续步骤。

步骤三、任务列表生成

根据当前时间节点T、映射函数各个任务的理论运行时间，统计在当前时间节点T时完成运行的任务，并将这些任务所属任务链的后继任务加入待调度任务列表，并继续后续步骤。

步骤四、搜索空间划分

对于待调度任务列表中每个任务生成所有可能的调度方案，并存入列表。

对于待调度任务列表中每个任务，根据被分配的内核id不同，任务的优先级不同，生成所有可能的调度方案，并存入列表中。根据调度方案的不同将递归搜索空间进行划分，每个调度方案将作为其所在的递归搜索空间的起始点，继续后续的搜索。

步骤五、调度方案搜索

对于搜索列表中的每个调度方案，各自分配一个线程，各个线程之间并行执行。对于每个线程，根据所述线程中的调度方案，更新中的映射关系。根据调度方案中为任务分配的优先级关系，执行优先级最高的任务，记录各个任务的开始执行时间。

同时，根据当前时间点，以及各个运行中任务的理论运行时间，计算下一个时间节点T，要满足至少存在一个任务将在T时完成执行。

在本发明的具体实施过程中，和T的更新可以并行进行。

基于不同调度方案各自的T和，各个线程并行地创建新的递归分支，开始递归搜索。

本发明还提供了一种高效实时调度系统，所述调度系统能够实现上述的实时调度方法，包括：DAG生成模块和任务调度模块。

所述DAG生成模块，其主要功能是解析用户提供的标准XML输入文件，并基于其中的信息构建有向无环图（DAG）实例，以供后续任务调度模块使用。XML输入文件详细描述了待调度的任务集合及其相互依赖关系，每个任务的关键信息包括任务名称、周期、执行时延、优先级以及后继任务。此外，输入文件还包含任务链的最大执行时间限制和任务起止时间约束，以支持更精确的调度控制。DAG生成模块在读取XML文件后，会提取并解析这些关键信息，构建任务节点及其依赖关系，并最终生成一个完整的DAG类实例。该DAG实例为任务调度模块提供清晰的任务拓扑结构，利于后续的调度分析与优化。

DAG模块由四个核心组成部分：DAG单元、DAGSignal单元、DAGTask单元和TaskPath单元，它们共同构成了DAG模块和任务调度系统的基础架构。

1）DAGTask（任务类）单元：该类用于表示系统中的任务，每个任务都有唯一的标识符，并具备名称、类型、周期、执行时延、优先级等关键信息。此外，任务可能受到最大执行时间（可选）和起止时间限制（可选）的约束。在任务执行完成后，会生成相应的信号（DAGSignal），该信号可用于触发其他任务的执行。

2）DAGSignal（信号类）单元：该类用于表示任务之间的依赖关系。在任务执行完成后，它会生成一个DAGSignal信号，该信号可以作为触发条件，使一个或多个后继任务开始执行。这种机制确保了任务执行的正确顺序，并支持复杂的任务依赖管理。

3）TaskPath（任务链类）单元：TaskPath表示任务执行链，它由多个DAGTask按照依赖关系顺序组织而成，确保任务按照正确的执行顺序进行。

4）DAG（有向无环图类）单元：DAG类是整个调度系统的核心，负责管理所DAGTask、DAGSignal和TaskPath实例，形成完整的任务拓扑结构。

所述任务调度模块，其核心功能是基于DAG类实例中存储的任务、任务链信息以及相关预处理数据，采用启发式搜索方法生成最优的任务调度方案。调度过程中，该模块会综合考虑任务的优先级、依赖关系以及时间约束等因素，动态调整调度策略，以满足系统的实时性要求。最终，调度模块的输出是一个数组，其中详细记录了每个任务的调度时间以及对应的CPU核心分配情况。该调度方案确保在满足所有约束条件的前提下（例如低优先级任务在其周期内完成、不超过最大执行时间限制、满足起止时间约束等），优先保证高优先级任务链的最优执行，即尽可能缩短关键任务链的整体执行时间，提高系统的整体调度效率。

任务调度模块由三部分组成：TimeNode单元、ResultNode单元和SimpleBacktrack单元。

1）TimeNode（任务执行时间类）单元：TimeNode与DAGTask一一对应，主要用于记录任务的开始执行时间（m\_startTime）和剩余执行时间（m\_restTime）。在任务调度过程中，当任务发生抢占（即一个更高优先级的任务中断当前任务的执行）时，TimeNode会记录被抢占任务的剩余执行时间，以便在后续恢复执行时能够准确地继续完成剩余的工作量。

2）ResultNode（任务调度结果类）单元：ResultNode与DAGTask同样一一对应，用于记录任务在当前启发式搜索空间中的调度状态，它包含任务的开始执行时间（m\_startTime）和执行优先级（m\_priority）。

3）SimpleBacktrack（启发式搜索类）单元：SimpleBacktrack类即用于启发式搜索最佳调度方案的类，它依赖于DAG类实例中的DAGTask类和TaskPath类来搜索最佳调度方案。在搜索过程中，任务的执行状态通过更新任务对应的TimeNode类来进行维护，任务当前的调度状态则由ResultNode类来进行维护。

本发明还提供了一种实现上述方法的硬件系统，所述硬件系统包括：存储器和处理器；所述存储器上存储有计算机程序，当所述计算机程序被所述处理器执行时，实现上述方法。

本发明还提供了一种计算机可读存储介质，其上存储有计算机程序，所述计算机程序被处理器执行时，实现上述方法。

本发明还提供了一种上述方法，上述系统，上述硬件系统或上述计算机可读存储介质在基于汽车开放系统架构的汽车操作系统在自动驾驶场景下从图像采集、预处理、特征提取到目标识别任务调度等中的应用。

本发明的有益效果包括：本发明的调度方法正式定义了多核平台中实时系统模型和具有资源约束的任务最优调度问题，为相关调度方法的设计提供了明确的数学基础；本发明提出的高效的实时任务调度方法，使任务链整体运行时间减少了约25%；利用并行策略和剪枝策略大幅减少搜索时间，减少了约95.9%。

**附图说明**

为了更清楚地说明本发明实施例或现有技术中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图做简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

图1是本发明调度系统的DAG生成模块结构图。

图2是本发明调度系统的任务调度模块结构图。

图3是本发明调度方法流程图。

图4是本发明实施例的任务链图。

图5是本发明实施例的任务参数与约束说明图。

图6是本发明实施例的任务链约束说明图。

图7是本发明实施例的调度有效性的实验结果展示图。

图8是本发明实施例的并行剪枝策略有效性的实验结果展示图。

**具体实施方式**

结合以下具体实施例和附图，对本发明作进一步的详细说明。实施本发明的过程、条件、实验方法等，除以下专门提及的内容之外，均为本领域的普遍知识和公知常识，本发明没有特别限制内容。

本发明正式定义了多核平台中实时系统模型和具有资源约束的任务最优调度问题，为算法设计提供了明确的数学基础；提出了一种面向多核平台的汽车操作系统任务的高效实时调度方法，通过模拟任务链的执行过程，递归搜索所有可能的调度方案，并利用并行策略和剪枝策略大幅减少搜索时间和计算复杂度；

本发明中的实时调度系统包括DAG生成模块和任务调度模块；

图1为本发明实时调度系统的DAG生成模块结构图，所述DAG生成模块基于输入的文件信息构建DAG实例，提供任务调度信息；

所述DAG生成模块进一步包括：作为调度系统核心的DAG单元、表示调度系统中任务的DAGTask单元、表示任务之间依赖关系的DAGSignal单元、表示任务执行链的TaskPath单元；

图2为本发明实时调度系统的任务调度模块，所述任务调度模块基于DAG类实例中存储的任务、任务链信息以及相关预处理数据，采用启发式搜索方法生成最优的任务调度方案；

所述任务调度模块进一步包括：记录任务执行时间的TimeNode单元、记录任务在当前启发式搜索空间中的调度状态的ResultNode单元、用于启发式搜索最佳调度方案的SimpleBacktrack单元。

如图3所示，本发明的实时调度方法包括以下步骤：

步骤一、终止条件设定

对于每条递归分支，在其起始阶段测试当前调度方案是否满足终止条件，所述终止条件即所有任务都已存在于当前搜索到的映射函数（即图3中的）中。若满足，且优于全局映射函数ap（即图3中的M），则在线程安全的前提下用当前搜索到的映射函数更新全局映射函数，并停止当前递归分支；若不满足，说明存在尚未调度的任务，继续后续步骤。

步骤二、中间结果剪枝

若递归搜索未满足终止条件，则根据当前已完成的部分任务的调度方案进行剪枝判断：计算理想运行时间，若按照当前调度方法，即使尚未调度的任务不考虑任何资源约束，其理想运行时间仍然没有优于全局最优调度结果，则进行剪枝；和/或，若当前调度方案违反任务链最大执行时长约束，则进行剪枝；和/或，若当前调度方案违反任务起止时间约束，则进行剪枝。若进行剪枝，则立刻停止当前递归分支；若不进行剪枝，则继续后续步骤。

步骤三、任务列表生成

根据当前时间节点T、映射函数各个任务的理论运行时间，统计在当前时间节点T时完成运行的任务，并将这些任务所属任务链的后继任务加入待调度任务列表，并继续后续步骤。

步骤四、搜索空间划分

对于待调度任务列表中每个任务生成所有可能的调度方案，并存入列表。

对于待调度任务列表中每个任务，根据被分配的内核id不同，任务的优先级不同，生成所有可能的调度方案，并存入列表中。根据调度方案的不同将递归搜索空间进行划分，每个调度方案将作为其所在的递归搜索空间的起始点，继续后续的搜索。

步骤五、调度方案搜索

对于搜索列表中的每个调度方案，各自分配一个线程，各个线程之间并行执行。对于每个线程，根据所述线程中的调度方案，更新中的映射关系。根据调度方案中为任务分配的优先级关系，执行优先级最高的任务，记录各个任务的开始执行时间。同时，根据当前时间点，以及各个运行中任务的理论运行时间，计算下一个时间节点T，要满足至少存在一个任务将在T时完成执行。基于各自的T和，各个线程并行地创建新的递归分支，开始递归搜索。

基于不同调度方案各自的T和，各个线程并行地创建新的递归分支，开始递归搜索。

本发明调度方法的具体算法如下：

|  |
| --- |
| 算法：基于并行剪枝的任务调度算法 |
| **Input：** 列表  ，任务链列表  任务列表  调度任务列表    **Output：**：最优映射函数 |
| 搜索  所有任务都已在 中被调度  （线程安全的条件下）{ 更优， 用 更新};  ;  计算理想运行时间并决定是否剪枝;  判断现有调度方案是否违反约束并决定是否剪枝；  **For** 每个任务 **in** **do**  **If** 在时间点T时已执行完成 **then**  将的所有后续任务添加到中;  记录的结束时间为;  为中的生成所有可能的计划，并保存在;  （以下并行执行）  **For** i=0 **to do**  **For** j=0 **to do**    将任务分配给核，并将其执行优先级设置为；  将映射添加到 中;  清空L;  所有核心执行具有最高优先级的任务，并将这些任务的开始时间以记录在中;  将的开始时间记录为;  更新;  搜索;  **End**  ; |

其中，本发明中的实时调度方法涉及到三个私有全局变量，包括用于存储最佳调度结果的全局映射函数Map，表示任务当前时间点的T，用于存储等待调度到核心的任务列表L。

核心列表表示多核平台所有可用的核心，其每个元素表示核心的 id。核心列表的大小是每个核心分配给要执行的任务队列的可用内核的数量。

所述任务链定义为，其中为任务链的任务序列；为任务链的循环周期时间，表示任务链将每隔时间就会被触发，任务开始依次进行执行；为任务链的被优先调度等级，在其他约束都满足的情况下，调度要求优先保证高调度等级任务链的总体运行时间最小；为任务链的最大执行时限，要求任务链的总体运行时间小于时间。对于任务序列，其中任务的顺序表示它们之间的数据依赖关系。对于两个相邻的任务，其中，当完成运行之后，其计算得出的数据结果将作为输入参数送入中，开始任务对于每个任务，都包含3个属性：理论运行时间，任务起始时刻和任务结束时刻，这要求在每一个循环周期中，都需要满足在时刻后开始运行，并在时刻前运行结束。

对于存储最佳调度结果的映射函数，如果任务被调度到核心，具有优先级，并从时间开始运行，则的值为 1，否则为 0。优先级确定调度在同一核心的确切位置的两个不同任务的执行顺序。四元组 可以表示任何任务的所有可能调度结果。

对于本发明中的资源约束任务最优调度问题：给定一个核心列表，一个集合表示所有的任务链，集合 表示任务链中所有的任务，则有具有资源约束的任务最优调度问题检索映射关系。

同时满足数据依赖约束、资源约束、最大执行时长约束、起止时间约束和目标优化约束。

（1）数据依赖约束：对于同一条任务链中的任意两个任务和，在任务链中位于之前。当两个任务都完成调度时，需要满足的任务开始时间加上其理论运行时间小于等于的任务开始时间，即保证必须在运行结束之后才开始运行。

形式化表示：, ,若

则

（2）资源约束：对于每一个内核，对于任意时刻允许最多一个任务在运行，若存在多个任务，则它们的调度优先级都不相同。

形式化表示：，有

则有，否则有种不同的优先级；

（3）最大执行时长约束**：**给定任何时刻，若任务链中存在任务没有被调度，即任务链尚未运行结束，则有减小于等于E，其中为任务链中的第一个任务的调度开始时间，E为该任务链的最大执行时长约束。

形式化表示：，

若有

则有

（4）起止时间约束**：**对于任意任务，其起止时间约束为和，若该任务已经完成调度，则其任务链中前置任务的理论运行时间之和需要小于等于，且考虑到同一个内核高优先级任务的抢占后，该任务执行完成的时间小于等于。

形式化表示：对，

若

则

且

（5）目标优化约束：

假设任务链的调度优先级为（按从大到小排列），则优化目标为依次使被调度优先级为的任务链的执行时长最短，即最小化

的值。

本发明能够面对多核平台上的自动驾驶系统，通过自动化的方式，快速、准确地生成针对其复杂任务链的最优调度方案，以克服传统手动设计方法在面对日益增长的设计复杂度时所面临的诸多局限性，能够在复杂的设计环境中，有效地提高任务调度的效率和质量，减少整体任务调度时间，同时具备良好的可扩展性和鲁棒性。

**实施例**

本发明所述方法关注于多核平台的汽车操作系统任务的高效实时调度，作为一次具体的实施例，过程如下：

1. 本发明采用一个基于汽车开发系统架构的真实自动驾驶系统的核心任务链作为实验用例。该自动驾驶系统的核心任务是对车载摄像头所拍摄的图像进行一系列复杂的处理，以实现对周围环境的精准感知与理解，从而为自动驾驶决策提供可靠的数据支持。

2. 图4详细展示了该系统的任务链图，其由13个任务和7个任务链构成，图5展示了这些任务的相关属性，包括任务的理论运行时间、运行周期和任务的起止时间约束，图6展示了这些任务链的最大执行时长约束和优先调度等级。实验用例涵盖了自动驾驶系统从图像采集、预处理、特征提取到目标识别等多个关键环节，这些任务链之间的相互协作与高效调度对于整个系统的实时性能至关重要。

3. 本实施例的自动调度代码部署于配备有Ubuntu操作系统（版本20.04）、8GB内存以及Intel i9-10900K CPU的高性能工作站，该CPU包含具备10个内核以及20个线程。

4. 为了精准评估所生成任务调度方案的运行时间性能，将生成的任务调度配置导入到基于汽车开发系统架构的TDA4目标平台上。该平台具备重放调度并测量汽车开发系统架构环境中系统实际运行时间开销的能力，记录各个任务链的实际运行时间。

5. 通过将本发明生成的最优任务调度设置方案与传统的人工手动调度设置方案分别导入目标平台中，测试并记录各个任务链的实际运行时间，进行深入的对比分析，以评估所提出方法在实际应用中的优势。传统的人工手动调度设置方案采用启发式的调度方案，即按照任务的运行先后顺序，依次将任务的调度至内核中，尽可能保证不存在空闲的内核。从实验结果中可以得出，本发明在所有任务链上均显著优于手动调度方案，实验结果如图7所示。具体而言，本发明得出的任务调度方案，能够实现更短的任务运行时间，整体运行时间减少了约25%，表明了本发明的有效性。

6. 为了验证并行策略在减少最优调度方案的搜索时间方面发挥的作用，修改搭载调度的工作站的可用线程数，利用不同数量的线程来执行并行策略，并详细记录不同的可用线程下，搜索到最优调度方案所需的时间开销，同时将其与基本调度方法进行对比分析。其中基本调度方法为单线程，时间开销为2920毫秒。实验结果如图8所示。从实验结果中可以得出，与未添加并行策略的基本调度方法相比，并行算法的加速比与所使用的线程数呈现出明显的相关性。当可用线程数为1时，具有并行策略的调度方法的时间开销与基本方法相同；当可用线程数为16时，具有并行策略的调度方法的时间开销是基本调度方法（单线程方法）开销的202/2920=6.92%。这为所提出并行策略的有效性提供了有力的实证支持，证明了其在提高计算效率方面的巨大潜力。

7. 为了验证剪枝策略在减少最优调度方案的搜索时间方面发挥的作用，分别部署应用剪枝策略与未应用剪枝策略的调度方法，对比搜索到最优调度方案所需时间开销，实验结果如图8所示。从实验结果中可以得出，剪枝策略有效地提高了最优调度方案的搜索速度。同时，剪枝策略的剪枝效率也随着可用线程数量的提高而增强。当可用线程数为1时，应用剪枝策略可以使得搜索开销降低12.6%；当可用线程数为16时，应用剪枝策略可以使得搜索开销降低40.7%。这为所提出剪枝策略的有效性提供了有力的实证支持，证明了其在提高计算效率方面的巨大潜力。

在所提出的并行策略和剪枝策略的有力辅助下，借助搜索线程之间的高效协同作用，整体搜索时间得以大幅缩短，减少了约95.9%。真实自动驾驶系统上的实验结果充分证实了所提出方法在调度时间以及所获得解决方案质量方面相较于手动导出的解决方案所展现出的显著优越性。这一成果不仅为自动驾驶系统中基于汽车开发系统架构的任务调度优化提供了全新的思路与方法，也为未来相关领域的研究与发展奠定了坚实的基础，具有重要的理论与实践价值。

本领域内的技术人员应明白，本申请的实施例可提供为方法、系统、或计算机程序产品。因此，本申请可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且，本申请可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质（包括但不限于磁盘存储器、CD-ROM、光学存储器等）上实施的计算机程序产品的形式。本申请实施例中的方案可以采用各种计算机语言实现，例如，面向对象的程序设计语言Java和直译式脚本语言JavaScript等。

本申请是参照根据本申请实施例的方法、设备（系统）、和计算机程序产品的流程图和／或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

尽管已描述了本申请的优选实施例，但本领域内的技术人员一旦得知了基本创造性概念，则可对这些实施例作出另外的变更和修改。所以，所附权利要求意欲解释为包括优选实施例以及落入本申请范围的所有变更和修改。

显然，本领域的技术人员可以对本申请进行各种改动和变型而不脱离本申请的精神和范围。这样，倘若本申请的这些修改和变型属于本申请权利要求及其等同技术的范围之内，则本申请也意图包含这些改动和变型在内。

本发明的保护内容不局限于以上实施例。在不背离本发明构思的精神和范围下，本领域技术人员能够想到的变化和优点都被包括在本发明中，并且以所附的权利要求书为保护范围。

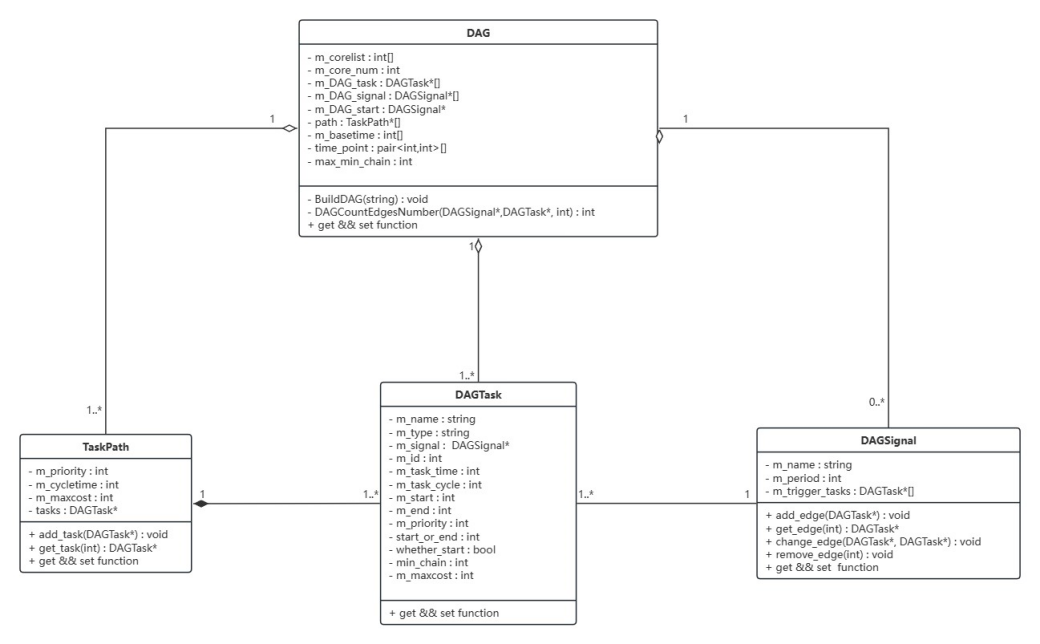


图1

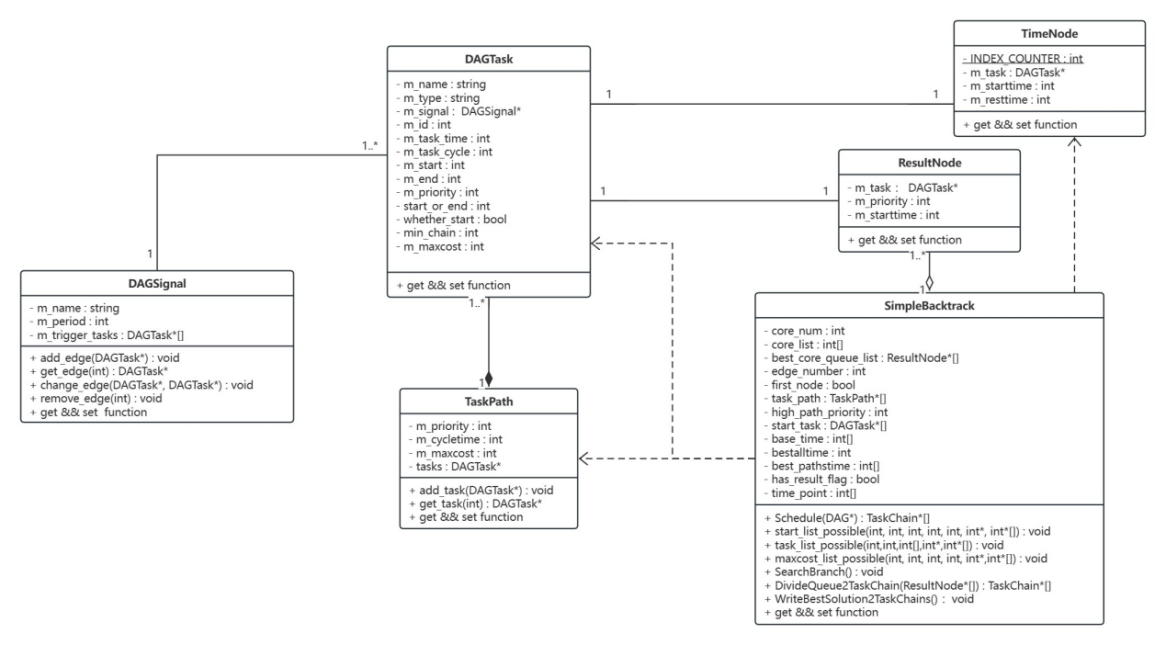


图2

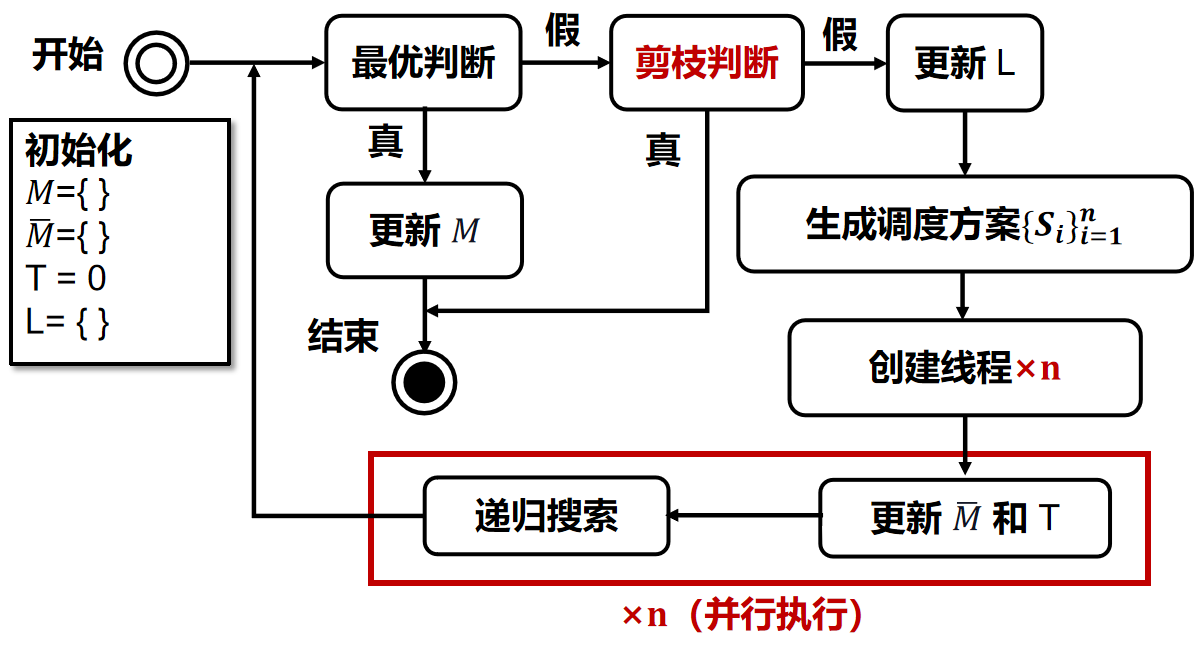


图3

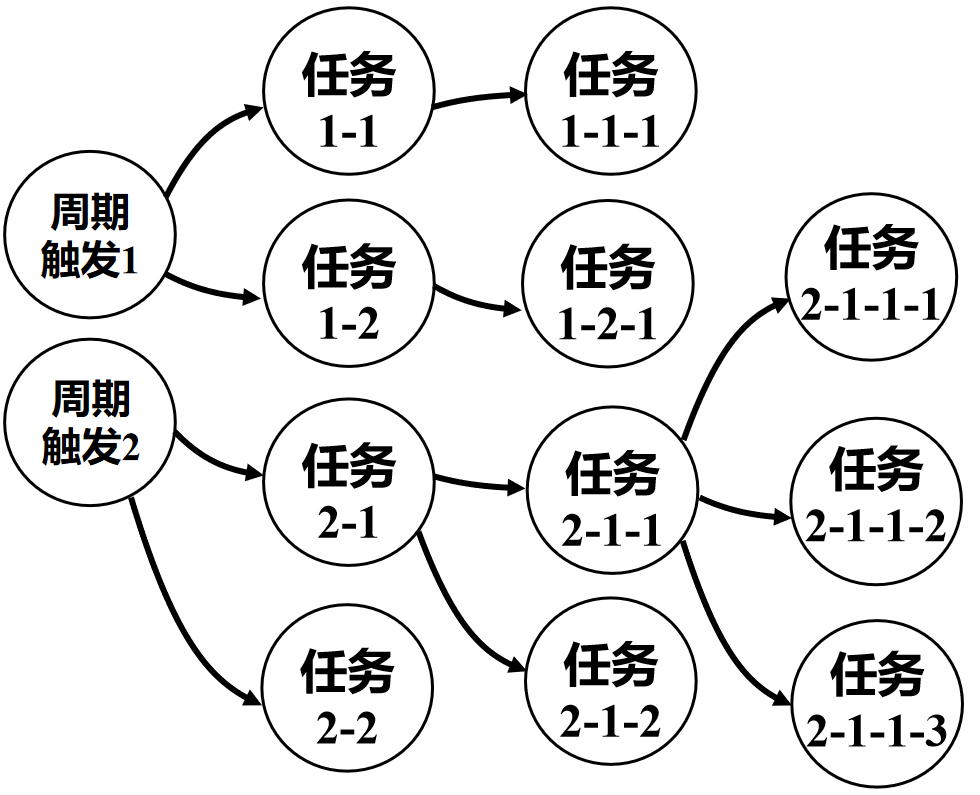


图4

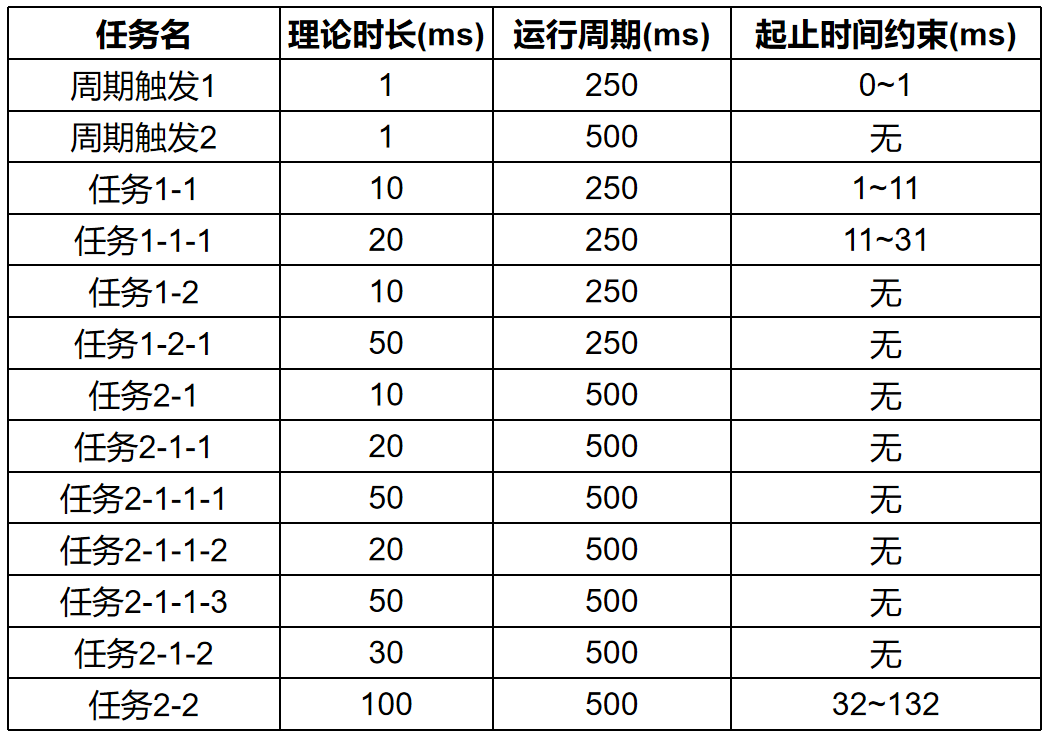


图5

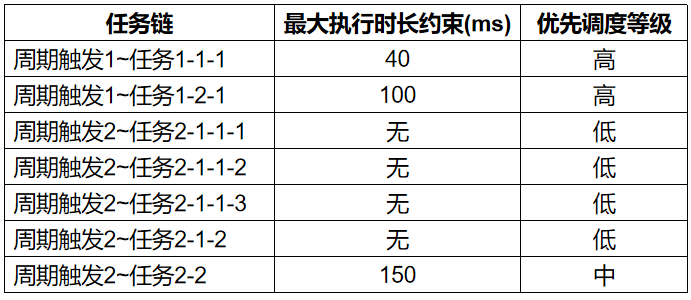


图6

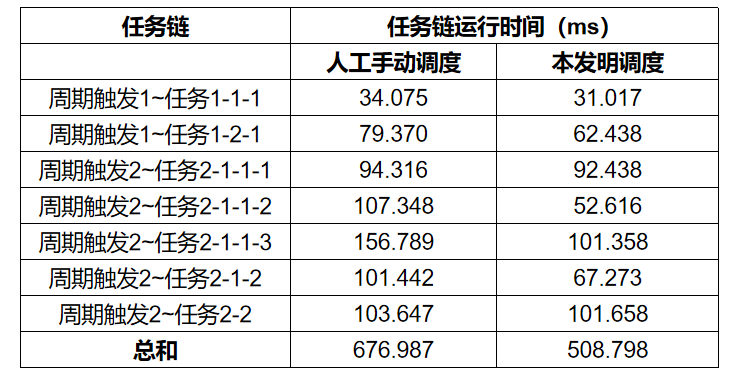


图7

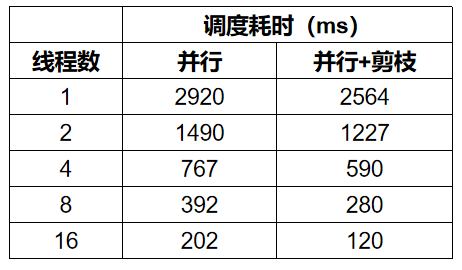


图8