(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 121833146 A
(43)申请公布日 2026.04.10

(21)申请号 202511423311.3

(22)申请日 2025.09.30

(30)优先权数据
24205298.3 2024.10.08 EP

(71)申请人 TTTech电脑科技公司
地址 奥地利

(72)发明人 彼得·马特尔
西尔维乌·克拉丘纳斯
拉蒙·塞尔纳奥利弗

(74)专利代理机构 北京集佳知识产权代理有限公司 11227
专利代理师 高岩

(51)Int.Cl.
G06F 9/48(2006.01)

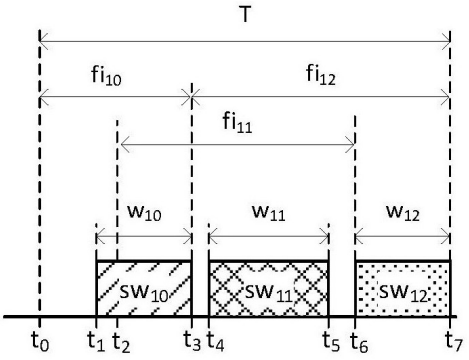
权利要求书5页 说明书19页 附图2页

(54)发明名称

在计算机系统中以灵活性执行间隔执行时间触发任务的方法

(57)摘要

一种根据全局时间触发的调度在计算机系统中执行任务集的作业的方法,其中,计算机系统包括在计算机系统的一个或更多个中央处理单元上的一个或更多个核,- 其中,所述全局时间触发的调度包括用于所述核中的每一个的一个时间触发的调度“核调度”,其中,任务集中的要在特定核上执行的任务根据所述核的核调度在所述核上执行。核调度为每个作业指定灵活性间隔,其中,作业的灵活性间隔的持续时间等于或长于所述作业的执行间隔,其中,一个作业的至少灵活性间隔长于所述作业的执行间隔,其中,作业的灵活性间隔不晚于所述作业的执行间隔的发布点开始,并且其中,作业的灵活性间隔不早于所述作业的执行间隔的完成点结束。



CN 121833146 A

1.一种根据全局时间触发的调度在计算机系统中执行任务集的作业的方法，

- 其中,所述计算机系统包括在所述计算机系统的一个或更多个中央处理单元上的一个或更多个核，

- 其中,所述全局时间触发的调度包括用于所述核中的每一个的一个时间触发的调度“核调度”,其中,所述任务集中的要在特定核上执行的任务根据所述核的核调度在所述核上执行，

- 其中,所述全局时间触发的调度为所述任务集指定具有循环长度的循环，

其中,核的核调度为所述任务集中的要在所述核上执行的每个任务指定以下：

●所述任务的每个任务实例“作业”的执行间隔,在所述执行间隔期间能够执行所述作业,其中,每个执行间隔包括所述执行间隔开始的发布点和所述执行间隔结束的完成点,其中,执行间隔的持续时间是所述执行间隔的所述发布点与所述完成点之间的时间跨度的持续时间,其中,作业的执行间隔限定了以下时间间隔:在所述时间间隔期间所述作业具有在所述核上执行的保证,以及

●周期,根据所述周期,在所述核上执行所述任务的作业，

○其中,所述任务的每个作业的执行间隔根据所述循环被循环地迭代,并且

其中,相同任务和不同任务中的任何两个作业的执行间隔不交叠,并且

其中,所述循环是所述任务集中的所有任务的所有周期的公倍数，

其特征在于：

所述核调度为每个作业指定灵活性间隔,其中,作业的灵活性间隔的持续时间等于或长于所述作业的执行间隔,其中,

至少一个作业的灵活性间隔长于所述作业的执行间隔,其中,

作业的灵活性间隔不晚于所述作业的执行间隔的发布点开始,并且其中,

作业的灵活性间隔不早于所述作业的执行间隔的完成点结束,其中,

作业的灵活性限定了能够执行所述作业的时间间隔,并且其中,

提供了运行时调派装置,所述运行时调派装置在针对每个核考虑在所述核调度中指定的所有灵活性间隔的情况下,根据所述全局时间触发的调度来选择和启动作业以供执行。

2.根据权利要求1所述的方法,其中,在所述任务集中的要在所述核上执行的那些任务的作业中,提供作业的至少一个第一子集,其中,对于所述至少一个第一子集中的作业,所述核调度指定彼此不交叠的灵活性间隔,其中,作业的灵活性间隔中的至少一个比所述作业的执行更长。

3.根据权利要求1或2所述的方法,其中,在所述任务集中的要在所述核上执行的那些任务中,提供作业的至少一个第二子集,其中,对于所述至少一个第二子集中的作业,所述核调度指定灵活性间隔,使得作业的所述至少一个第二子集中的两个或更多个作业的灵活性间隔相同,即,对于所述至少一个第二子集中的所述两个或更多个作业,所述两个或更多个作业的灵活性间隔的开始时刻是相同的时刻,并且所述两个或更多个作业的灵活性间隔的结束时刻是相同的时刻。

4.根据权利要求1至3中任一项所述的方法,其中,在所述任务集中的要在所述核上执行的那些任务中,提供了作业的至少一个第三子集,其中,所述至少一个第三子集中的任何两个作业的灵活性间隔部分地交叠。

5.根据权利要求2至4中任一项所述的方法,其中,对于任务作业的所述至少一个第一子集中的作业,所述运行时调派装置根据以下规则进行动作:

■如果当前时间在作业的灵活性间隔内,则:

○如果用于所述作业的核空闲,即没有作业正在执行,则所述作业将被启动以执行;以及

○如果用于所述作业的核不空闲,即另一作业正在执行,则所述选择的作业将在以下中首先出现的时刻处被启动以执行:

■用于所述作业的核变为空闲的时刻,或者

■所述当前时间到达所述选择的作业的执行间隔的开始。

6.根据权利要求3至5中任一项所述的方法,其中,对于任务作业的所述至少一个第二子集中的作业,所述运行时调派装置根据以下规则进行动作:

■如果所述当前时间在一个或多个灵活性间隔内,则将基于以下限定的准则之一来执行对所述灵活性间隔相等的作业中的一个作业的选择:

○将选择和启动具有最早执行间隔的作业以执行;或者

○将选择和启动具有最短执行时间的作业以执行;或者

○将选择和启动由基于限定的约束例如最早截止期限或最高优先级进行选择而得到的作业以执行;或者

○将选择以下的概率更高的作业:在比作业的执行间隔中考虑的时间更少的时间内完成所述作业的执行,从而提供更多的松弛即未使用的CPU执行时间。

7.根据权利要求6所述的方法,其中,被选择并启动以执行的所述作业将保持执行,直到发生以下事件中的任何一个:

■所述作业完成所述作业在所述当前循环内的执行,或者

■所述作业达到与所述作业的执行间隔对应的执行时间量,而不管所述当前时间在所述作业的执行期间在所述灵活性间隔内的相对位置如何,优选地加上所述执行间隔内可用的任何松弛量,被计算为:

○所述灵活性间隔内的现有空闲间隔的和,被计算为所述灵活性间隔的长度与所述灵活性间隔内的所有作业的执行间隔的和之间的差,或者

○由在所述灵活性间隔内执行的任何作业提供且未被任何其他先前作业消耗的松弛的和,

或者

■所述作业达到与所述作业的执行间隔对应的执行时间量加上限定的超支预算,被指定为:

○固定时间间隔,例如1 ms,或者

○所述作业的自身执行时间的相对百分比,例如10%,或者

■所述当前时间到达所述灵活性间隔的结束。

8.根据权利要求4至7中任一项所述的方法,其中,对于作业的所述至少一个第三子集中的作业,所述运行时调派装置根据以下规则进行动作:

■当所述当前时间到达灵活性间隔的开始时,与所述灵活性间隔有关的作业将被添加到合格作业的列表,并且其中,

■当满足以下条件之一时：

○以下核变为空闲:所述作业要在所述核上执行，

○当前执行的作业完成所述作业的执行,或者被停止,或者被强制停止，

○所述当前时间到达所述合格作业的列表中的作业之一的执行间隔的开始，

所述运行时调派装置将从所述合格作业的列表中选择以下作业:对于该作业,该作业的执行间隔的开始较早地发生。

9.根据权利要求8所述的方法,其中,在以下情况下,作业的执行被停止或者所述作业被强制停止执行，

●当所述当前时间等于所述作业的执行间隔的完成点(完成时刻)并且在所述合格作业的列表中存在以下作业时:对于该作业,该作业的执行间隔的开始等于所述当前时间,或者

●当所述当前时间晚于所述作业的执行间隔的完成点并且在所述合格作业的列表中存在以下作业时:对于该作业,该作业的执行间隔的开始点(开始时刻)等于所述当前时间,或者

●当所述当前时间到达所述执行的作业的灵活性间隔的结束时,如果在所述合格作业的列表中没有以下作业:对于该作业,该作业的执行间隔的开始等于所述当前时间。

10.一种运行时调派装置,用于根据全局时间触发的调度来选择任务的作业并且启动所述作业以在计算机系统中执行,特别是用于在根据权利要求1至9中一项所述的方法中使用，

- 其中,所述计算机系统包括在所述计算机系统的一个或更多个中央处理单元上的一个或更多个核，

- 其中,所述全局时间触发的调度包括用于所述核中的每一个的一个时间触发的调度“核调度”,其中,所述任务集中的要在特定核上执行的任务的作业根据所述核的核调度在所述核上执行，

- 其中,所述全局时间触发的调度为所述任务集指定具有循环长度的循环，

其中,核的核调度为所述任务集中的要在所述核上执行的每个任务的作业指定以下：

●所述任务的每个任务实例“作业”的执行间隔,在所述执行间隔期间能够执行所述作业,其中,每个执行间隔包括所述执行间隔开始的发布点和所述执行间隔结束的完成点,其中,执行间隔的持续时间是所述执行间隔的所述发布点与所述完成点之间的时间跨度的持续时间,其中,作业的执行间隔限定了以下时间间隔:在所述时间间隔期间所述作业具有在所述核上执行的保证,以及

●周期,根据所述周期,在所述核上执行所述任务的作业，

○其中,所述任务的每个作业的执行间隔根据所述循环被循环地迭代,并且

其中,相同任务和不同任务中的任何两个作业的执行间隔不交叠,并且

其中,所述循环是所述任务集中的所有任务的所有周期的公倍数，

其特征在于：

所述运行时调派装置被配置成在考虑到所述核调度的所有灵活性间隔的情况下,根据所述核调度来选择和启动任务的作业以供执行。

11.根据权利要求10所述的运行时调派装置,其中,所述运行时调派装置被配置成针对

作业的第一子集进行动作,对于所述作业的第一子集,所述核调度指定彼此不交叠的灵活性间隔,其中,根据以下规则,作业的灵活性间隔中的至少一个比所述作业的执行更长:

■如果当前时间在作业的灵活性间隔内,则:

○如果用于所述作业的核空闲,即没有作业正在执行,则所述作业将被启动以执行;以及

○如果用于所述作业的核不空闲,即另一作业正在执行,则所述选择的作业将在以下中首先出现的时刻处被启动以执行:

■用于所述作业的核变为空闲的时刻,或者

■所述当前时间到达所述选择的作业的执行间隔的开始。

12.根据权利要求10或11所述的运行时调派装置,其中,所述运行时调派装置被配置成针对作业的第二子集进行动作,对于所述作业的第二子集,所述核调度指定灵活性间隔,使得作业的所述第二子集中的两个或更多个作业的灵活性间隔相同,即,根据以下规则,对于所述第二子集中的所述两个或更多个作业,所述两个或更多个作业的灵活性间隔的开始时刻是相同的时刻,并且所述两个或更多个作业的灵活性间隔的结束时刻是相同的时刻:

■如果所述当前时间在一个或多个灵活性间隔内,则将基于以下限定的准则之一来执行对所述灵活性间隔相等的作业中的一个作业的选择:

○将选择和启动具有最早执行间隔的作业以执行;或者

○将选择和启动具有最短执行时间的作业以执行;或者

○将选择和启动由基于限定的约束例如最早截止期限或最高优先级进行选择而得到的作业以执行;或者

○将选择以下的概率更高的作业:在比作业的执行间隔中考虑的时间更少的时间内完成所述作业的执行,从而提供更多的松弛即未使用的CPU执行时间。

13.根据权利要求12所述的运行时调派装置,其中,所述运行时调派装置被配置成使得被选择和启动以执行的所述作业将保持执行,直到发生以下事件中的任何一个:

■所述作业完成所述作业在所述当前循环内的执行,或者

■所述作业达到与所述作业的执行间隔对应的执行时间量,而不管所述当前时间在所述作业的执行期间在所述灵活性间隔内的相对位置如何,优选地加上所述执行间隔内可用的任何松弛量,被计算为:

○所述灵活性间隔内的现有空闲间隔的和,被计算为所述灵活性间隔的长度与所述灵活性间隔内的所有作业的执行间隔的和之间的差,或者

○由在所述灵活性间隔内执行的任何作业提供且未被任何其他先前作业消耗的松弛的和,

或者

■所述作业达到与所述作业的执行间隔对应的执行时间量加上限定的超支预算,被指定为:

○固定时间间隔,例如1 ms,或者

○所述作业的自身执行时间的相对百分比,例如10%,

或者

■所述当前时间到达所述灵活性间隔的结束。

14.根据权利要求10至13中任一项所述的运行时调派装置,其中,所述运行时调派装置被配置成针对作业的第三子集进行动作,对于所述作业的第三子集,所述核心调度指定所述第三子集中的任何两个作业的灵活性间隔根据以下规则部分地交叠:

■当所述当前时间到达灵活性间隔的开始时,与所述灵活性间隔有关的作业将被添加到合格作业的列表,并且其中,

■当满足以下条件之一时:

○以下核变为空闲:所述作业要在所述核上执行,

○当前执行的作业完成所述作业的执行,或者被停止,或者被强制停止,

○所述当前时间到达所述合格作业的列表中的作业之一的执行间隔的开始,

所述运行时调派装置将从所述合格作业的列表中选择以下作业:对于该作业,该作业的执行间隔的开始较早地发生。

15.根据权利要求14所述的运行时调派装置,其中,在以下情况下,作业的执行被停止或者所述作业被强制停止执行:

●当所述当前时间等于所述作业的执行间隔的完成点并且在所述合格作业的列表中存在以下作业时:对于该作业,该作业的执行间隔的开始等于所述当前时间,或者

●当所述当前时间晚于所述作业的执行间隔的完成点并且在所述合格作业的列表中存在以下作业时:对于该作业,该作业的执行间隔的开始等于所述当前时间,或者

●当所述当前时间到达所述执行的作业的灵活性间隔的结束时,如果在所述合格作业的列表中没有以下作业:对于该作业,该作业的执行间隔的开始等于所述当前时间。

16.一种计算机系统,包括一个或更多个中央处理单元上的一个或更多个核,其中,所述计算机系统被配置成根据基于权利要求1至9中的一项所述的方法执行任务,所述计算机系统包括根据权利要求10至15中的一项所述的运行时调派装置。

17.一种生成用于在计算机系统的核上执行任务的作业的核调度的方法,特别是用于在根据权利要求1至9中的一项所述的方法中使用,其中,所述核调度包括用于要在所述核上执行的所述作业的灵活性间隔,所述方法包括以下步骤:

●为要在所述核上执行的所述任务提供时间触发的调度,所述时间触发的调度包括用于每个任务的每个作业的执行间隔和每个任务的周期,根据所述周期循环地执行所述任务的作业,

●基于所述时间触发的调度以及基于调度约束集来确定可调度性界限,从而产生每个执行间隔的上界限和下界限,

●例如,通过从根据用于所述核的所述时间触发的调度的任务的第一作业的执行间隔开始迭代地增加所述执行间隔,同时检查是否满足所述调度约束,以及

●一旦在迭代中至少一个调度约束未满足,则将先前迭代的扩展的执行间隔确定为所述第一作业的灵活性间隔,

●被选择为第一作业,直到所有作业都被提供了灵活性间隔,

选择下一作业并重复所述方法,直到所有作业都被提供了灵活性间隔。

在计算机系统中以灵活性执行间隔执行时间触发任务的方法

技术领域

[0001] 本发明涉及根据全局时间触发的调度在计算机系统中执行任务集的作业的方法，

[0002] - 其中，计算机系统包括在计算机系统的一个或更多个中央处理单元上的一个或更多个核，

[0003] - 其中，所述全局时间触发的调度包括用于所述核中的每一个的一个时间触发的调度“核调度”，其中，任务集中的要在特定核上执行的任务根据所述核的核调度在所述核上执行，

[0004] - 其中，全局时间触发的调度为任务集指定具有循环(cycle)长度的循环，

[0005] 其中，核的核调度为任务集中的要在所述核上执行的每个任务指定以下：

[0006] • 任务的每个任务实例“作业”的执行间隔，在该执行间隔期间可以执行作业，其中，每个执行间隔包括执行间隔开始的发布点和执行间隔结束的完成点，其中，执行间隔的持续时间是执行间隔的发布点与完成点之间的时间跨度的持续时间，其中，作业的执行间隔限定了以下时间间隔：在所述时间间隔期间作业具有在核上执行的保证，以及

[0007] • 周期(period)，根据该周期，在核上执行所述任务的作业，

[0008] o其中，任务的每个作业的执行间隔根据循环被循环地迭代，并且

[0009] 其中，相同任务和不同任务中的任何两个作业的执行间隔不交叠，并且

[0010] 其中，循环是任务集中的所有任务的所有周期的公倍数。

背景技术

[0011] 遵循时间触发范式的软件任务的周期性作业的执行已经在文献中进行了研究，并且在工业中已经采用了许多年。在它的益处中，突出的是以循环方式确定性地及时执行软件任务的作业，通常由允许设计即正确(correctness-by-design)方法的离线计算的调度(tt调度)驱动。此外，tt调度可以基于通过其他范式难以达到的优化目标来计算，如最小化作业级抖动或满足复杂的作业相互依赖性。

[0012] 然而，还已知的是，时间触发范式在动态适应软件作业的运行时行为的变化方面提供了很少灵活性或没有灵活性。随着时间的推移，这种适应性的缺乏可能是需要精确和确定性行为的关键系统的资产，但它也可能降低在具有不太严格确定性要求的系统中最大化资源的使用的能力。

[0013] 在计算tt调度之前，必须表征软件任务，特别是关于它们的及时性要求，如所需的执行时间和执行速率(即，周期)。时间触发范式的共同目标是基于最坏情况的及时性界限(如最坏情况的执行时间和最低执行速率)来计算tt调度。一旦构建了tt调度，它就在运行时通过调派机制循环地实施。考虑最坏情况保证了在运行时期间任务的作业的行为的任何变化将不会超过系统容量，并且因此仍然保证了执行的正确性。

[0014] 然而，在现代计算机系统特别是汽车计算机系统中，目标通常是使硬件部件(如CPU和存储器)的利用率最大化，因为任何未使用的资源将优选地在设计时按比例缩小以降低生产成本和整体功耗。

[0015] 这两个目标常常彼此冲突,并且无法简单地解决。事实上,向时间触发范式增加灵活性使得能够适应任务的周期性作业的执行及时性的运行时变化,同时保留确定性和正确性保证,这是一项挑战。

[0016] 时间触发的调度

[0017] 现有的时间触发调度方法通过根据限定的调度约束集来计算所述CPU核中的一个或多个CPU核中的所述软件任务的每个周期性作业的一个或多个执行间隔来计划软件任务在所限定的CPU核中的执行。

[0018] 执行间隔包括开始时间和持续时间,在持续时间内分配的CPU核将排他地执行所述软件任务的分配作业。

[0019] 计算的tt调度通常以时间表的形式表示,该时间表包含具有作业和CPU核的分配的计算间隔的列表。

[0020] tt调度的计算基于限定的调度约束,限定的调度约束确定了符合有效tt调度的执行间隔之间的正确关系。例如,执行间隔不能分配给多个作业,因为这将导致单个资源(CPU核)需要同时执行两个作业的指令,这是不可行的。作为另一示例,同一CPU核上的不同执行间隔不能交叠,因为这将再次导致多个作业计划在同一CPU核上同时执行,因此使每个单独作业的执行时间的保证的保留无效。

[0021] TT调度约束

[0022] 计算机系统包括一个或更多个主机,其中的每个主机包括一个或更多个CPU核。

[0023] 软件应用由一个或更多个周期性软件组件或任务组成。每个任务由其周期、最早发布时间和截止期限来表征——任务根据其周期在核上循环地执行,承载任务执行实例或作业的序列。

[0024] tt调度是根据文献(例如[1])中发现的TT调度的一般正确性约束来计算的,并且出于完整性,在此处进行了概述。tt调度的计算涉及在限定的调度循环内计算软件任务的每个周期性作业的执行间隔,其中,所述计算符合限定的调度约束,限定的调度约束包括但不限于:

[0025] 开始约束:软件任务sw_i的任何作业的执行的开始必须在每个周期p_i中的给定的最早发布时间e_i之后。

[0026] 截止期限约束:软件任务sw_i的任何作业的执行的结束必须在每个周期p_i中的给定的截止期限d_i之前。

[0027] 非交叠约束:任何任务的两个作业不能在同一核上同时执行。

[0028] 依赖性约束:进一步限制软件任务的相互关系的其他要求在计算tt调度时引入了附加的依赖性约束。例如,在一些情况下,功能正确性要求第一软件任务的作业的执行应当在所有循环中发生在第二软件任务的作业的执行之后。在其他情况下,一组软件任务的作业的执行顺序可以符合有向无环图(DAG),由此在某个最大时间间隔内,DAG的第一个软件任务和最后软件任务的作业应当完成它们的执行。取决于所述相互关系的复杂性,这些附加依赖性约束可以被分类为简单依赖性约束或复杂依赖性约束。

[0029] 任务可以具有简单的依赖性或复杂的依赖性。依赖性包括两个或更多个任务,其中,指定所述任务的作业之间的限定的执行顺序,其中,

[0030] • 在简单的依赖性中,所述包括的任务中的所有任务具有相同的周期,而

[0031] • 在复杂的依赖性中,所述包括的任务中的至少两个任务具有不同的周期。

[0032] 此外,简单的依赖性和复杂的依赖性涉及最大时延约束,其中,所述最大时延可以是以下中之一:

[0033] • 最大响应时间,指定对于限定的任务集中的第一任务的所有作业,存在通过该集中的任务的所有其他中间作业遵循限定的顺序的后继作业的跟踪,使得该跟踪花费的时间不长于所述响应时间,

[0034] • 最大数据龄期,指定对于根据所限定的顺序的任务集中的最后任务的所有作业,存在从任务集中的第一任务的一些作业经过该集中的所有其他中间任务的作业的跟踪,由此所述跟踪花费的时间不长于所述数据龄期。

[0035] 抖动约束:对于任务的每个作业,在调度的开始与最早发布时间之间存在非负的延迟;抖动约束限定了任务的任何一对作业的所述延迟之间的最大允许差异。

[0036] 已知的是,tt调度的构建是计算密集型操作,其对于大型系统而言,无法在合理的时间内被最佳地解决[2,3]。近似启发式算法在过去已被使用,并且在平均情况下表现得相当好[4]。

[0037] 运行时执行可变性

[0038] 在一些情况下,任务的作业序列可以例如由于它们需要处理的数据随时间的差异而沿它们的循环执行迭代表现出不同的运行时执行要求。在通过假设软件任务的每个作业的最坏情况运行时(最坏情况计算时间、或WCET),因此计划足以满足所述最坏情况行为(例如,最大执行路径)的执行间隔的量和长度来计算tt调度时,可以解决该运行时可变性。

[0039] 时间触发调度范式的优点是软件任务的执行作业在运行时期间的确定性和及时行为,而不管它们在执行要求中的循环变化如何,通常仅导致循环之间的最小偏差(例如,由于由底层硬件和软件组件的不确定性引入的不可避免的运行时不准确性)。

[0040] 运行时适应性

[0041] 在一般情况下,在tt调度的设计时间期间的构建防止了对软件任务的作业的动态可变性执行运行时适应,并且可能是时间触发调度范式的缺点。例如,当软件任务的作业不需要循环内的完整的执行间隔时,剩余时间(松弛)不能容易地重新分配给tt调度中的下一个软件任务的作业。

[0042] 时间触发调度范式的另一相关缺点是当计算时间表时,通过假设软件任务的时间性(特别是WCET)的最坏情况界限引起的过度配置,因此,分配比执行期间所需的平均值更大的执行间隔,这是因为在运行时期间,作业通常需要比它们各自的任务的WCET更少的执行时间。

[0043] 一些现有方法通过考虑下界限来计算tt调度,例如对于某些任务的平均执行时间而不是WCET[5],解决了由于WCET而引起的过度配置的负面影响,由此如果所述任务的作业在其循环中的任何循环中超过它们的分配的执行间隔,则它们可能被中断并且不能在当前循环中完成它们的执行,或者在一些情况下,由于超过它们的计划的执行间隔而损害其他软件任务的作业。

[0044] 其他现有方法允许计算机系统在运行时期间在替选的预先计算的tt调度(例如,被称为“模式”)集之间交换其配置[6]。这些方法试图在设计时估计将在运行时期间观察到的可变性,并且计算适应于这些条件的专用tt调度。配置的这种适应通常被称为执行系统

模式改变。在这些方法中,支持的模式的数量受到两个因素的限制:一方面,由于需要不同的tt调度而导致的可能的系统状态(可观察的可变性)的数量随着系统的大小(例如,CPU核的数量、软件任务的数量、约束的数量)指数地增长,并且因此这样的可能的系统状态中的仅一部分可以可行地存储在计算机系统中以供将来使用(例如,由于有限的存储器容量);另一方面,系统模式改变不能即时地被应用,并且可能需要复杂的算法来保证在转换期间运行时处的系统稳定性,因此增加了时延并限制了在部署的运行系统中的适应性机会。

[0045] 然而,其他现有方法包括在运行时期间响应于系统的实际状态而动态地修改计算的tt调度的机制[9,10]。这些方法受到通常在一个或几个执行循环内对新tt调度的快速(重新)计算中涉及的约束的计算复杂性的限制。因此,所采用的算法不能处理大量的约束,或者在足够快的响应时间内满足所有约束。此外,当动态变化可能负面地影响与安全关键任务(例如在部署的汽车或工业应用中的那些任务)有关的作业的及时执行时,安全和认证的忧虑出现。因此,在运行时处修改时间表在真实世界用例中具有有限的用途。

[0046] 操作系统/运行时系统调派器

[0047] 在计算机系统中提供活动模块,通常是软件组件,该模块通过在限定的时刻从就绪作业集中选择任务的作业并将它们调派用于在计算机系统中的CPU核中的一个或更多个中执行来编排软件任务的执行。该活动模块即调派器(或调派装置)是施行由调度器做出的决策的运行时组件,并且可以以不同形式例如作为独立的运行时软件组件、作为操作系统的模块或作为高级软件堆栈在计算机系统中实现。

[0048] 通常,提供一个这样的活动模块。然而,也可以提供两个或更多个这样的活动模块。

[0049] 在实现时间触发范式的计算机系统中,调派器基于tt调度选择作业以供执行。一旦作业被调派,调派器就等待直到所述作业的执行间隔结束,并且遵循tt调度(即,时间表)中限定的顺序和定时继续选择下一个作业。

[0050] 在实现其他调度范式的计算机系统中,调派器可以基于限定的准则来选择软件任务的作业以供执行。例如,在实现较早截止期限优先(EDF)范式的计算机系统中,调派器从就绪作业集中选择具有最近截止期限的作业以供执行。相反,在实现固定优先级(FP)范式的计算机系统中,软件任务由限定的和固定的优先级来表征,由此调派器从就绪作业集中选择具有最高限定优先级的软件任务的作业以供执行。当产生经调派的作业或新作业准备就绪时,调派器可以基于相同的准则来选择并启动新作业。

发明内容

[0051] 本发明的目的是在计算机系统中优化资源的使用,特别是CPU时间和/或核时间的使用,在该计算机系统中,周期性任务的作业根据时间触发的调度来执行。

[0052] 这是用根据权利要求1所述的方法实现的,其中,核调度为每个作业指定灵活性间隔,其中,作业的灵活性间隔的持续时间等于或长于所述作业的执行间隔,其中,

[0053] 一个作业的灵活性间隔至少长于所述作业的执行间隔,其中,

[0054] 作业的灵活性间隔不晚于所述作业的执行间隔的发布点(开始点、开始/发布时刻)开始,并且其中,

[0055] 作业的灵活性间隔不早于所述作业的执行间隔的完成点(结束点、结束/完成时

刻)结束,其中,

[0056] 作业的灵活性限定了在其期间可以执行所述作业的时间间隔,并且其中,

[0057] 提供了运行时调派装置,该运行时调派装置在针对每个核考虑如在核调度中指定的所有灵活性间隔的情况下,根据全局时间触发的调度来选择和启动作业以供执行。

[0058] 调度循环内的任务的周期性执行包括由时间表中的一个所述元素表示的所述任务的任务实例或作业的一个或更多个执行,其中,每个作业对应于根据其周期性对任务的执行,根据调度循环循环地重复。如果任务周期等于调度循环,则在调度循环内存在所述任务的单个作业,而如果调度循环大于该周期,则根据周期与调度循环之间的关系存在所述任务的多个作业。注意,调度循环的长度优选为所有任务的周期的公倍数,例如最小公倍数。例如,如果任务的周期是10并且循环是20,则在循环内将存在任务的两个实例("作业")。它们中的每一个将在其相应的周期间隔(从0..10到10..20)内具有其自己的间隔,并且灵活性间隔甚至可以具有不同的长度并且属于三个子集中的不同的一个,这些子集将在下面说明。

[0059] 根据本发明,为每个作业提供灵活性间隔,由此作业的灵活性间隔与该作业的执行间隔完全交叠并且等于或大于其执行间隔。这使得运行时调派装置能够在时间方面更灵活地组织作业的执行,并且从而更好地利用可用资源,特别是允许更好地利用每个核的时间。

[0060] 如所提及的,本发明要求至少一个作业的灵活性间隔长于所述作业的执行间隔。为了本发明的最佳效果,有利的是,若干作业、特别是多个作业、优选地所有作业的灵活性间隔长于对应的执行间隔。

[0061] 在一般情况下,一种计算tt调度的方法涉及针对要调度的每个作业的每个执行间隔决定在时间线上的有效放置,这经受限定的调度约束,所述限定的调度约束限定所述间隔的正确放置的可调度性界限(bounds)。一旦计算出tt调,所述约束由符合调度的所提供的执行间隔隐式地满足。每个作业的执行间隔的放置由限定的调度约束所施加的限制界定。然而,在计算过程中,在所述界限内多个决定是可能的,从而产生替选但同等正确的tt调度。

[0062] 例如,如果周期性和截止期限是要满足的唯一约束,则可以在由周期开始和相对截止期限界定的间隔内任意地调度经受截止期限约束的周期性作业。通过执行确定作业的执行间隔的决定,调度过程为所述作业选择多个可能的正确执行间隔中之一。

[0063] 在一些情况下,在计算tt调度期间做出的早期决定影响与其他作业有关的后期决定。例如,当作业经受依赖性约束时,影响在前作业的执行间隔的任何决定影响后继作业的决策,因为依赖性要求后继作业仅可以在前置作业的执行间隔之后被调度,因此将下界限紧缩至后继作业的执行间隔。

[0064] 计算tt调度的一些方法在决定每个作业的执行间隔的放置时,应用简单的策略来基于下界限选择执行间隔的开始,即尽可能快地调度作业。替选地,其他方法应用策略以在决定每个作业的执行间隔的放置时,基于上界限选择执行间隔的结束,即,尽可能晚地调度作业。更复杂的方法基于附加限定的准则,在决定每个作业的执行间隔的放置时选择所述界限之间的任意值。

[0065] 可以提供的是,通过在每个调度决定时迭代地调整所有作业的上界限和下界限,

来扩展计算tt调度的方法以保持跟踪和更新每个作业的可调度性界限。还可以提供的是，所述扩展的方法在计算tt调度之后，基于所述可调度性上界限和下界限附加地计算每个作业的灵活性间隔，其中，对于每个作业，其灵活性间隔的开始对应于可调度性下界限，并且其灵活性间隔的结束对应于可调度性上界限。

[0066] 还可以提供的是，一种基于现有tt调度和调度约束集来得出可调度性界限的方法，其中，所述方法计算每个执行间隔的上界限和下界限，这经受所限定的调度约束集。例如，迭代方法可以启动第一作业从tt调度到与其执行间隔对应的间隔的灵活性间隔，并且迭代地增加灵活性间隔，同时检查仍然满足所限定的约束集。一旦至少一个约束未被满足，所述第一作业的灵活性间隔被固定为先前迭代的灵活性间隔，并且下一作业被选择为第一作业，直到所有作业都被提供了灵活性间隔。

[0067] 可以提供的是，在任务集中的要在核上执行的那些任务的作业中，提供作业的至少一个第一子集，其中，对于所述至少一个第一子集中的作业，核调度指定彼此不交叠的灵活性间隔，其中，作业的灵活性间隔中的至少一个比所述作业的执行更长。

[0068] 其中包括灵活性间隔的作业是具有遵循所述第一子集的规则的灵活性间隔的作业的该第一种情况最大限度地简化了调派器的运行时行为。

[0069] 替选地或附加地，可以提供的是，在任务集中的要在核上执行的那些任务中，提供作业的至少一个第二子集，其中，对于所述至少一个第二子集中的作业，核调度指定灵活性间隔，使得作业的所述至少一个第二子集中的两个或更多个作业的灵活性间隔相同，即，对于所述至少一个第二子集中的所述两个或更多个作业，所述两个或更多个作业的灵活性间隔的开始时刻是相同的时刻，并且所述两个或更多个作业的灵活性间隔的结束时刻是相同的时刻。例如，作业的灵活性间隔中的至少一个比所述作业的执行更长。

[0070] 其中包括灵活性间隔的作业是具有遵循所述第二子集的规则的灵活性间隔的作业的该第二种情况也允许调派器的简化运行时行为，因为可以提供的是，该第二子集的作业可以在所述间隔内以任何任意顺序执行。

[0071] 替选地或附加地，可以提供的是，在任务集中的要在核上执行的那些任务中，提供了作业的至少一个第三子集，其中，所述至少一个第三子集中的任何两个作业的灵活性间隔部分地交叠。例如，作业的灵活性间隔中的至少一个比所述作业的执行更长。

[0072] 其中包括灵活性间隔的作业是具有遵循所述第三子集的规则的灵活性间隔的作业的该第三种情况给运行时调派装置增加了最大复杂性。

[0073] 在该上下文中，注意，所描述的子集是自包含的，这意味着对于具有交叠间隔的那些作业，交叠必须在同一集的作业之间(即，不与处于其他集之一中的作业交叠)。

[0074] 非交叠子集(第一子集，子集1)是最具限制性的子集：对于要处于该子集中的作业，其灵活性间隔不应当与该子集中的任何其他作业的任何灵活性间隔交叠。如果在两个作业的灵活性间隔之间存在交叠，则两个交叠作业不是第一子集的一部分。

[0075] 接下来是具有相同灵活性间隔的第二子集(子集2)。该子集的作业的灵活性间隔应当仅与具有相同灵活性间隔的其他作业的灵活性间隔交叠。如果存在与不相同的灵活性间隔的交叠，则两个作业也不能在第二子集中。

[0076] 灵活性间隔的任何其他组合指的是第三子集的任务。

[0077] 在本发明中，必须存在来自至少一个子集的任务，其中，所述至少一个存在的子集

包含比对应的执行间隔更长的至少一个灵活性间隔。然而,也可能存在来自两个或甚至三个子集的作业。调派装置必须被配置成能够处理来自存在的子集的作业。通常,提供了可以处理来自所有子集的作业的调派器,但是调派装置还可以包括用于当前子集中的每一个的单独的调派器。

[0078] 应当注意的是,可以存在每个类型的更多的子集。例如,可以存在“类型”子集1的两个不同子集,一个具有3个作业,一个具有5个作业,其中,第一子集1的所有3个作业彼此交叠,并且第二子集1的所有5个作业彼此交叠,但是3个作业中没有一个与5个作业中的任何一个交叠。

[0079] 关于由调派装置应用的调派策略,下面将描述与上述作业的子集(子集1至子集3)连接的不同的可能情况(情况1至情况3)。

[0080] 为了完整起见,此处描述了可能发生的“情况0”,因为仍然可以存在具有“刚性”执行间隔而没有灵活性间隔的作业,这样的作业也必须由调派装置处理。在这种情况下的调派策略如下:运行时调派装置将按时间表迭代,同时保持跟踪与循环有关的时间进度。调派装置(也被称为“调派器”)将当前时间与限定每个执行间隔的开始和结束的时刻进行比较,这些时刻将在时间表中递增地排序。以下规则描述了情况0中的运行时调派器的行为:

[0081] • 在执行间隔开始时,相应的作业将被选择并启动以执行。如果另一个作业正在执行,则它将被抢占(即,中断)。

[0082] • 在所述执行间隔结束时,如果作业仍在运行,则将停止作业并且将如上所述的那样选择新作业。

[0083] • 如果在时间间隔(例如,时间表中的间隙)内不存在执行间隔,则调派器将屈服(yield)并等待直到下一个执行间隔。

[0084] • 如果作业在其执行间隔结束之前完成执行,则调派器将屈服。

[0085] 根据情况1(非交叠的灵活性间隔),可以提供的是,对于任务作业的至少一个第一子集的作业,运行时调派装置根据以下规则进行动作:

[0086] ■如果当前时间在作业的灵活性间隔内,则:

[0087] o如果用于所述作业的核空闲,即没有作业正在执行,则所述作业将被启动以执行;以及

[0088] o如果用于所述作业的核不空闲,即另一作业正在执行,则所述选择的作业将在以下中首先出现的时刻处被启动以执行:

[0089] ■用于所述作业的核变为空闲的时刻,或者

[0090] ■当前时间到达所选择的作业的执行间隔的开始。

[0091] 根据该情况1的运行时调派器将按时间表迭代,同时保持跟踪与循环有关的时间进度。调派器将对当前时间与限定每个灵活性间隔的开始和结束的时刻进行比较,这些时刻将在时间表中递增地排序。

[0092] 在非空闲(处理)核的情况下,作业将在任何情况下在其执行间隔的发布点处开始。在灵活性间隔的结束处,如果作业仍在运行,则作业将被停止。如果该时刻对应于另一作业的灵活性间隔的开始,则将如上所述的那样选择所述作业。

[0093] 如果在时间间隔(例如,时间表中的间隙)内不存在灵活性间隔,则调派器(=运行时调派装置)将屈服,直到下一个灵活性间隔。屈服是指放弃对CPU或核的控制,因为调派器

无事可做(不存在现在可以执行的作业)。

[0094] 如果所有作业都在其灵活性间隔结束之前完成执行,则调派器将屈服,直到下一个灵活性间隔。

[0095] 通常,执行间隔限定了保证的间隔,但它被用于选择和启动作业以供其执行的灵活性间隔所取代。因此,作业的执行时间可以大于它们的执行间隔的时间,这在以下情况中是优点:任务的确切最坏情况执行时间未知时,或者对于所述任务的一个或所有作业被低估时,或者作业最终可能超限时。

[0096] 此外,在一般情况下,可能存在另一个调派器正在控制和选择其他任务的其他作业以在tt调度的"间隙"中(即,当tt调度"屈服"时)执行。因此,当根据本发明的调派器调度tt调度中的任务的作业时,可能已经存在"另一个"任务的作业在运行,"另一个"任务可以是与tt调度有关的任务集中的一个,或者是与某个其他调派器有关的某个其他任务。

[0097] 当这样的附加调派器(例如分层地)在计算机系统中活动时,其他任务(例如固定优先级任务)的作业的执行可以在所选择的作业的灵活性间隔内发生,至多直到执行间隔的开始。

[0098] 在这种情况下,本发明的优点是:

[0099] ■允许其他任务(例如,固定优先级任务)的作业在不损害由tt调度提供的任务的保证执行间隔的情况下执行,以及

[0100] ■允许tt调度中的任务的作业一旦处理核在其灵活性间隔内保持空闲时但不迟于执行间隔开始的时刻执行,以及

[0101] ■允许tt调度中的任务的作业执行,直到其灵活性间隔结束。

[0102] 根据情况2(严格交叠的灵活性间隔),可以提供的是,对于任务作业的至少一个第二子集中的作业,运行时调派装置根据以下规则进行动作:

[0103] ■如果当前时间在一个或多个灵活性间隔内,则将基于以下限定的准则之一来执行对所述灵活性间隔相等的作业之一的选择:

[0104] o将选择和启动具有最早执行间隔的作业以执行;或者

[0105] o将选择和启动具有最短执行时间的作业以执行;或者

[0106] o基于限定的约束(例如,最早截止期限或最高优先级)的选择得到的作业将被选择并启动以执行(这允许选择基于动态算法如最早截止期限优先(EDF)或固定优先级调度器),或者

[0107] o将选择以下的概率更高的作业:在比作业的执行间隔中考虑的时间更少的时间内完成作业的执行,从而提供更多的松弛即未使用的CPU执行时间。

[0108] 根据该情况2的运行时调派装置将按时间表迭代,同时保持跟踪与循环有关的时间进展。调派器将对当前时间与限定每个灵活性间隔的开始和结束的时刻进行比较,这些时刻将在时间表中递增地排序。该子情况下的选择基于对作业将留下未使用的执行时间的概率的估计。然后,该时间松弛可以由共享灵活性间隔界限的作业的子集内的其他超限作业使用。可以在过去循环中的执行期间动态地测量作业较早完成的概率,例如通过计算一系列过去执行循环上的移动平均值,或者它可以基于静态分析并在运行时之前预定义。

[0109] 在情况2的上下文中,可以提供的是,被选择和启动以执行的作业将保持执行,直到以下事件中的任何一个发生:

[0110] ■作业完成作业在当前循环内的执行,或者

[0111] ■作业达到与该作业的执行间隔对应的执行时间量,而不管当前时间在作业的执行期间在灵活性间隔内的相对位置如何,优选地加上执行间隔内可用的任何松弛量,被计算为:

[0112] o灵活性间隔内的现有空闲间隔的和,被计算为灵活性间隔的长度与所述灵活性间隔内的所有作业的执行间隔的和之间的差(这可以在运行时之前静态地确定;该条件保证不会留下具有少于与其执行间隔的长度对应的间隔的作业),或者

[0113] o由在灵活性间隔内执行的任何作业提供且未被任何其他先前作业消耗的松弛的和(这可以基于作业的执行动态地确定),

[0114] 或者

[0115] ■作业达到与该作业的执行间隔对应的执行时间量加上限定的超支预算,被指定为:

[0116] o固定时间间隔,例如1 ms,或者

[0117] o该作业的自身执行时间的相对百分比,例如10%,

[0118] 或者

[0119] ■当前时间到达灵活性间隔的结束。

[0120] 选择过程将对剩余的作业(即,排除已经执行的作业)重复如上。如有必要,将更新松弛量和统计参数。

[0121] 一旦所有作业完成了其在该循环内的执行,则调派器将屈服,直到灵活性间隔结束,并且然后重复。

[0122] 根据最一般的情况,即情况3,不同作业的灵活性间隔可以与其他作业的灵活性或执行间隔部分地交叠。注意,不同作业的执行间隔不能交叠。根据该情况,运行时调派器将按时间表迭代,同时保持跟踪与循环有关的时间进度。调派器将对当前时间与限定每个灵活性间隔和执行间隔的开始和结束的时刻进行比较,这些时刻将在时间表中递增地排序。

[0123] 在该上下文中,可以提供的是,对于作业的至少一个第三子集中的作业,运行时调派装置根据以下规则进行动作:

[0124] ■在当前时间到达灵活性间隔的开始时,与所述灵活性间隔有关的作业将被添加到合格作业的列表,并且其中,

[0125] ■当满足以下条件之一时:

[0126] o将在其上执行作业的核变为空闲,

[0127] o当前执行的作业完成该作业的执行,或者被停止,或者被强制停止,

[0128] o当前时间到达合格作业的列表中的作业之一的执行间隔的开始,

[0129] 运行时调派装置将从合格作业的列表中选择以下作业:对于该作业,其执行间隔的开始较早地发生。

[0130] 一旦从合格作业列表中选择了作业,所述作业就从所述列表中移除并被启动以供执行。当合格作业的列表为空时,运行时调派装置将屈服。

[0131] 还可以提供的是,在以下情况下,作业的执行被停止或者作业被强制停止执行,

[0132] • 在当前时间等于所述作业的执行间隔的完成点并且在合格作业的列表中存在该作业的执行间隔的开始等于当前时间的作业时,或者

[0133] • 在当前时间晚于所述作业的执行间隔的完成点并且在合格作业的列表中存在该作业的执行间隔的开始等于当前时间的作业时,或者

[0134] • 在当前时间到达所述执行的作业的灵活性间隔的结束时,如果在合格作业的列表中没有该作业的执行间隔的开始等于当前时间的作业。

[0135] 在该上下文中,应当注意,可以提供的是,在满足上述条件之前作业通过其自身完成其执行。

[0136] 根据本发明,存在情况1至3中的至少一个和调派装置的对应策略。然而,也可能的是,其他情况,也是情况0,或所有情况0、1至3存在于调度中。调派装置被配置成处理存在的所有那些情况,结合或不结合情况0中描述的策略来实现情况1、2和3中描述的运行时调派策略中的至少一个。

[0137] 此外,本发明涉及一种运行时调派装置,用于根据全局时间触发调度来选择任务的作业并启动所述作业以在计算机系统中执行,特别是用于在如上所述的方法中使用,

[0138] - 其中,计算机系统包括在计算机系统的一个或更多个中央处理单元上的一个或更多个核,

[0139] - 其中,所述全局时间触发调度包括用于所述核中的每一个的一个时间触发调度“核调度”,其中,任务集中的要在特定核上执行的任务的作业根据所述核的核调度在所述核上执行,

[0140] - 其中,全局时间触发的调度为任务集指定具有循环长度的循环,

[0141] 其中,核的核调度为所述任务集中的要在所述核上执行的每个任务的作业指定以下:

[0142] • 任务的每个任务实例“作业”的执行间隔,在该执行间隔期间可以执行作业,其中,每个执行间隔包括执行间隔开始的发布点和执行间隔结束的完成点,其中,执行间隔的持续时间是执行间隔的发布点与完成点之间的时间跨度的持续时间,其中,作业的执行间隔限定了以下时间间隔:在所述时间间隔期间所述作业具有在核上执行的保证,以及

[0143] • 周期,根据该周期,在核上执行所述任务的作业,

[0144] o其中,任务的每个作业的执行间隔根据循环被循环地迭代,并且

[0145] 其中,相同任务和不同任务中的任何两个作业的执行间隔不交叠,并且

[0146] 其中,循环是任务集中的所有任务的所有周期的公倍数。

[0147] 运行时调派装置的特征在于,运行时调派装置被配置成在考虑到核调度的所有灵活性间隔的情况下,根据核调度来选择和启动任务的作业以供执行。

[0148] 本发明涉及运行时调派装置,例如,运行时调派器模块,例如操作系统调派器,负责在要在核上执行的作业中选择一个作业,并且启动所述作业以执行。所述选择在作业集的合格作业子集(通常称为就绪作业集)中执行,并且取决于以上提及的情况与每个作业的相应间隔有关。

[0149] 在一些情况下,运行时调派器可以实现多个调派策略,或者在其他情况下,多个调派器可以同时共存于计算机系统中。例如,系统可以由两个调派器组成,一个调派器实现固定优先级策略,并且一个调派器实现时间触发策略,其中,每个所述调派器涉及单独的作业集。

[0150] 一些计算机系统,例如基于GNU/Linux的计算机系统,实现了分层组织的若干运行

时调派器。当顶部调派器在其任务集的合格作业列表中没有作业可供选择以执行时,顶部调派器屈服并委托层次结构中的下一个调派器来执行选择。如果没有调派器具有合格的作业供选择,则相应的核(CPU)空闲,直到作业变得合格。

[0151] 结合情况1,可以提供的是,运行时调派装置被配置成针对作业的第一子集进行动作,针对该作业的第一子集,核调度指定彼此不交叠的灵活性间隔,其中,作业的灵活性间隔中的至少一个比根据以下规则对所述作业的执行更长:

[0152] ■如果当前时间在作业的灵活性间隔内,则:

[0153] o如果用于所述作业的核空闲,即没有作业正在执行,则所述作业将被启动以执行;以及

[0154] o如果用于所述作业的核不空闲,即另一作业正在执行,则所述选择的作业将在以下中首先出现的时刻处被启动以执行:

[0155] ■用于所述作业的核变为空闲的时刻,或者

[0156] ■当前时间到达所选择的作业的执行间隔的开始。

[0157] 结合情况2,可以提供的是,运行时调派装置被配置成针对作业的第二子集进行动作,针对该作业的第二子集,核调度指定灵活性间隔,使得作业的所述第二子集中的两个或更多个作业的灵活性间隔相同,即,根据以下规则,对于所述第二子集中的所述两个或更多个作业,所述两个或更多个作业的灵活性间隔的开始时刻是相同的时刻,并且所述两个或更多个作业的灵活性间隔的结束时刻是相同的时刻:

[0158] ■如果当前时间在一个或多个灵活性间隔内,则将基于以下限定的准则之一来执行对所述灵活性间隔相等的作业之一的选择:

[0159] o将选择和启动具有最早执行间隔的作业以执行;或者

[0160] o将选择和启动具有最短执行时间的作业以执行;或者

[0161] o将选择和启动由基于限定的约束例如最早的截止期限或最高的优先级进行选择而得到的作业以执行;或者

[0162] o将选择以下的概率更高的作业:在比作业的执行间隔中考虑的时间更少的时间内完成作业的执行,并且提供更多的松弛即未使用的CPU执行时间。

[0163] 结合情况2,还可以提供,运行时调派装置被配置成使得被选择和启动以执行的作业将保持执行,直到发生以下事件中的任何一个:

[0164] ■作业完成作业在当前循环内的执行,或者

[0165] ■作业达到与该作业的执行间隔对应的执行时间量,而不管当前时间在作业的执行期间在灵活性间隔内的相对位置如何,优选地加上执行间隔内可用的任何松弛量,被计算为:

[0166] o灵活性间隔内的现有空闲间隔的和,被计算为灵活性间隔的长度与所述灵活性间隔内的所有作业的执行间隔的和之间的差,或者

[0167] o由在灵活性间隔内执行的任何作业提供且未被任何其他先前作业消耗的松弛的和,

[0168] 或者

[0169] ■作业达到与该作业的执行间隔对应的执行时间量加上限定的超支预算,被指定为:

[0170] o固定时间间隔,例如1 ms,或者

[0171] o该作业的自身执行时间的相对百分比,例如10%,

[0172] 或者

[0173] ■当前时间到达灵活性间隔的结束。

[0174] 结合情况3,可以提供的是,在以下情况下,运行时调派装置被配置成针对作业的第三子集进行动作,针对该作业的第三子集,核心调度指定所述第三子集中的任何两个作业的灵活性间隔根据以下规则部分地交叠:

[0175] ■在当前时间到达灵活性间隔的开始时,与所述灵活性间隔有关的作业将被添加到合格作业的列表,并且其中,

[0176] ■当满足以下条件之一时:

[0177] o要在其上执行作业的核变为空闲,

[0178] o当前执行的作业完成该作业的执行,或者被停止,或者被强制停止,

[0179] o当前时间到达合格作业的列表中的作业之一的执行间隔的开始,

[0180] 运行时调派装置将从合格作业的列表中选择以下作业:对于该作业,其执行间隔的开始较早地发生。

[0181] 结合情况3,还可以提供的是,在以下情况下,作业的执行被停止或者作业被强制停止执行:

[0182] •在当前时间等于所述作业的执行间隔的完成点并且在合格作业的列表中存在该作业的执行间隔的开始等于当前时间的作业时,或者

[0183] •在当前时间晚于所述作业的执行间隔的完成点并且在合格作业的列表中存在该作业的执行间隔的开始等于当前时间的作业时,或者

[0184] •在当前时间到达所述执行的作业的灵活性间隔的结束时,如果在合格作业的列表中没有该作业的执行间隔的开始等于当前时间的作业。

[0185] 本发明还涉及一种计算机系统,该计算机系统包括一个或更多个中央处理单元上的一个或更多个核,其中,计算机系统被配置成根据如上所述的方法执行任务,计算机系统包括如上所述的运行时调派装置。

[0186] 此外,本发明涉及一种生成用于在计算机系统的核上执行任务的作业的核调度的方法,特别是用于在如上所述的方法中使用的方法,其中,核调度包括要在所述核上执行的作业的灵活性间隔,该方法包括以下步骤:

[0187] •为要在所述核上执行的任务提供时间触发的调度,该时间触发的调度包括用于每个任务的每个作业的执行间隔和每个任务的周期,根据该周期循环地执行所述任务的作业,

[0188] •基于所述时间触发的调度以及基于调度约束集来确定可调度性界限,从而产生每个执行间隔的上界限和下界限,

[0189] •例如,通过从根据用于所述核的所述时间触发的调度的任务的第一作业的执行间隔开始迭代地增加所述执行间隔,同时检查调度约束是否被满足,以及

[0190] •一旦在迭代中未满足至少一个调度约束,则将先前迭代的扩展的执行间隔确定为所述第一作业的灵活性间隔,

[0191] •被选择为第一作业,直到所有作业都被提供了灵活性间隔,

[0192] 选择下一作业并重复所述方法,直到所有作业都被提供了灵活性间隔。

附图说明

[0193] 参照附图以非限制性方式描述本发明:

[0194] 图1描绘了针对没有灵活性间隔的作业集TS_1的时间触发的调度。

[0195] 图2描绘了针对具有非交叠灵活性间隔的作业集TS_2的具有灵活性间隔的时间触发的调度。

[0196] 图3描绘了针对具有完全交叠的灵活性间隔的作业集TS_3的具有灵活性间隔的时间触发的调度。

[0197] 图4描绘了针对具有部分交叠的灵活性间隔的作业集的具有灵活性间隔的时间触发的调度。

[0198] 图5至图8描绘了根据本发明的实施方式的图1至图4中描绘的时间触发的调度的示例性执行实例。

具体实施方式

[0199] 计算机系统,例如汽车计算机系统,包括一个或更多个主机,特别是处理器,其中所述主机集$H = \{h_1..h_n\}$中的每个主机h_i包括一个或更多个CPU核,其由CPU核集$C^H = \{c^h_1..c^h_k\}$来表征,其中,所述CPU核被配置成执行指令。软件应用例如汽车功能包括软件组件(任务),所述软件组件包括可以在所述一个或多个CPU核中执行的指令序列。

[0200] 软件任务在所述CPU核中循环地执行,其中,在所限定的软件任务集$SW = \{sw_1..sw_m\}$中的每个软件任务sw_i的特征在于,

[0201] • 其执行循环的长度,即其周期,或最小到达间隔(interarrival)时间p_i,以及

[0202] • 其最大所需执行预算w_i,其对应于在一个循环内在一个CPU核中执行所述任务所需的累积时间,例如其最坏情况计算时间(WCET),或者以及所需计算时间的估计。

[0203] 软件任务sw_i还可以由以下来表征,

[0204] • 其最早发布时间e_i,其指示软件任务相对于循环开始准备执行的最早相对时间,以及/或者

[0205] • 其执行截止期限d_i,其指示软件任务应当完成与所述循环对应的执行的循环开始之后的最新相对时间点。

[0206] CPU中的软件任务遵循时间触发调度范式来执行,由此所述计算机系统的运行时组件(调派器),例如操作系统的运行时调派器组件选择并调派所述任务(作业)的实例,以遵循预定义的时间触发的调度(tt调度)在所述CPU核中按限定的时间间隔或间隔来执行。

[0207] 每个所述间隔由开始时间或时间偏移o和持续时间或长度b来表征。

[0208] 用于CPU核c的tt调度,即“核调度”,涉及时间表TT^c,其由表循环z^c'的长度、或调度循环、以及一个或更多个元素的序列来表征,其中,所述时间表的每个所述元素由对来自软件任务SW集的软件任务sw的分配作业的引用来表征。

[0209] 在调度循环内的任务的周期性执行包括由时间表中的一个所述元素表示的所述任务的任务实例或作业的一个或更多个执行,其中,每个作业对应于任务根据其在调度循环内的周期性的执行。如果任务周期等于调度循环,则在调度循环内存在所述任务的单个

作业,而如果调度循环大于该周期,则根据周期与调度循环之间的关系存在所述任务的多个作业。注意,调度循环的长度应当是所有任务的周期的公倍数,例如,最小公倍数。

[0210] 在下面示出的示例中,为了简单起见,假设每个特定任务在具有循环长度T的循环内仅执行一次(确切地说,在循环内仅执行所述任务的一个作业),使得在下面的描述中,为了简单起见,使用措辞“任务的执行”而不是措辞“任务的作业的执行”。

[0211] 根据本发明,所述时间表的每个元素还由附加的灵活性间隔fi来表征。

[0212] 表1示出了示例性任务集(也表示为“任务集”)TS_0,其包括三个任务(每个任务一个作业)sw_1、sw_2和sw_3,其特征在于它们的周期p、早期发布时间e(最早开始点)、截止期限d(最晚完成点)和执行预算w。

[0213]

	sw_1	sw_2	sw_3
p	T	T	T
e	t_0	t_2	t_5
d	t_2	t_5	t_7
w	w_1	w_2	w_3

[0214] 表1

[0215] 参考符号“T”表示由全局时间触发调度指定的循环长度。

[0216] 图1描绘了根据现有技术的由离线的时间触发调度器针对所述任务集TS_0计算的时间触发的调度的示例。如从图1可以看出,与现有技术中的情况一样,向每个任务分配了固定的执行间隔,并且不提供灵活性间隔。

[0217] 表2以时间表的形式表征来自图1的tt调度,其中,调度循环z等于周期p,(即,z=T)并且执行间隔是s_1、s_2和s_3(o为开始时间,b为持续时间/长度),

[0218]

	o	b	sw
s_1	t_1	$t_2 - t_1$	sw_1
s_2	t_3	$t_4 - t_3$	sw_2
s_3	t_5	$t_6 - t_5$	sw_3

[0219] 表2

[0220] 表3示出了示例性任务集(“第一”任务集)TS_1,其包括三个任务sw_4、sw_5和sw_6,其特征在于它们的周期p、早期发布时间e、截止期限d和执行预算w。

[0221]

	sw_4	sw_5	sw_6
p	T	T	T
e	t_0	t_2	t_5
d	t_2	t_5	t_7
w	w_4	w_5	w_6

[0222] 表3

[0223] 图2描绘了用于来自表3的任务集TS_1的时间触发的调度,对于任务集TS_1中的任务,其具有执行间隔s_4、s_5、s_6和非交叠的灵活性间隔fi_4、fi_5和fi_6。

[0224] 表4以时间表的形式表征来自图2的tt调度,其中,调度循环等于周期T(即,z=T),并且执行间隔是s_4、s_5和s_6。参数o和b是指相应任务的执行间隔,fi表示相应任务的灵活性间隔。

[0225]

	o	b	sw	fi
s_4	t_1	$t_2 - t_1$	sw_4	$fi_4 = (t_0, t_2)$
s_5	t_3	$t_4 - t_3$	sw_5	$fi_5 = (t_2, t_5)$
s_6	t_5	$t_6 - t_5$	sw_6	$fi_6 = (t_5, t_7)$

[0226] 表4

[0227] 表5示出了示例性任务集(“第二”任务集)TS_2,其包括三个任务sw_7、sw_8和sw_9,其特征在于它们的周期p、早期发布时间e、截止期限d和执行预算w。

[0228]

	sw_7	sw_8	sw_9
p	T	T	T
e	t_0	t_0	t_0
d	t_7	t_7	t_7
w	w_7	w_8	w_9

[0229] 表5

[0230] 图3描绘了用于对于任务集TS_2具有相同的、完全交叠的灵活性间隔fi_7、fi_8和fi_9的任务集的具有灵活性间隔的时间触发的调度。

[0231] 表6以时间表的形式表征来自图3的tt调度,其中,调度循环等于周期T(即,z=T),并且执行间隔(参数o、b)是s_7、s_8和s_9。

[0232]

	o	b	sw	Fi
s_7	t_1	$t_2 - t_1$	sw_7	$fi_7 = (t_0, t_7)$
s_8	t_3	$t_4 - t_3$	sw_8	$fi_8 = (t_0, t_7)$
s_9	t_5	$t_6 - t_5$	sw_9	$fi_9 = (t_0, t_7)$

[0233] 表6

[0234] 包括三个任务sw_{10}、sw_{11}和sw_{12}的附加示例性任务集(“第三”任务集)TS_3在表7中由任务集TS_3中的每个任务sw_{10}、sw_{11}、sw_{12}的周期p、早期发布时间e、截止期限d和执行预算w来表征。

[0235]

	sw_{10}	sw_{11}	sw_{12}
p	T	T	T
e	t_0	t_0	t_0
d	t_7	t_7	t_7
w	w_{10}	w_{11}	w_{12}

[0236] 表7

[0237] 图4描绘了用于对于来自表7的任务集TS_3的具有部分交叠的灵活性间隔fi_{10}、fi_{11}和fi_{12}的任务集的具有灵活性间隔的时间触发的调度。

[0238] 表8以时间表的形式表征来自图4的tt调度,其中,调度循环等于周期T(即,z=T),并且执行间隔(参数o、b)是s_{10}、s_{11}、s_{12}。

[0239]

	o	b	sw	fi
s_{10}	t_1	$t_3 - t_1$	sw_{10}	$fi_{10} = (t_0, t_3)$
s_{11}	t_4	$t_5 - t_4$	sw_{11}	$fi_{11} = (t_2, t_6)$
s_{12}	t_6	$t_7 - t_6$	sw_{12}	$fi_{12} = (t_3, t_7)$

[0240] 表8

[0241] 图5至图8描绘了图1至图4中描绘的并且在上述表中表征的时间触发的调度的示例性执行实例(“作业”)。

[0242] 情况0:(现有技术)

[0243] 如上所述,示例性任务集TS_0包括三个任务sw_1、sw_2和sw_3。

[0244] 时间触发调度器相应地计算用于如表2中所示的没有灵活性间隔的TS_0的时间触发的调度。

[0245] 出于该示例的目的,图5描绘了所述任务集TS_0的可能执行轨迹,其中,任务集中的

每个任务的任务执行时间小于或等于任务的对应执行间隔。

[0246] 每个任务的开始执行时间被严格地界定到由时间触发的调度确定的执行间隔的开始,而不管对应的时间触发的调度中的任何先前任务的提前终止如何。例如,图5中的sw_2的执行在时间t_3处开始,与任务sw_1已经在相对于如时间触发调度表2中限定的任务sw_1的执行间隔t_2的结束的较早时间点t_{11}处完成执行的事实无关。

[0247] 此外,根据没有本发明的附加优点的一般时间触发调派器,图5中的任务sw_3的执行时间的结束被上界定为执行间隔t_6的结束,而不管该任务在该点t_6处是否已经完成其执行。

[0248] 情况1:

[0249] 包括三个任务sw_4、sw_5和sw_6的第一任务集TS_1在上面详细地描述并在表3中被表征。包括本发明的优点的时间触发调度器可以计算如表4中表征和图2中描绘的时间触发的调度,其具有用于不交叠的任务的灵活性间隔。

[0250] 出于该示例的目的,图6描绘了所述第一任务集TS_1的可能执行轨迹,其中,任务集中的一些任务的任务可用执行时间大于对应的任务执行间隔。

[0251] 如图6中所描绘的,根据本发明的时间触发调派器(调派装置)可以选择并启动任务以供执行。例如,在示例性轨迹中,如果CPU核在sw_4的执行的执行间隔t_1的开始之前的时间t_0是空闲的,则图6中的sw_4的执行可以在该时间处开始(由于其灵活性间隔)。

[0252] 在该示例中,任务sw_5在其灵活性间隔期间在晚于其执行间隔s_5(=t_4-t_3)的开始实例t_3的t_{31}处开始,并且晚于其执行间隔的结束t_4(但在其灵活性间隔的结束之前)结束。因此,根据图6的任务sw_5的执行时间t_{31}的开始既不被界定于其灵活性间隔t_2的开始,也不被界定于其执行间隔t_4的开始,从而例如允许CPU核执行图中未描绘的其他任务,或者保持处于低功率模式以节省能量。

[0253] 图6中的任务sw_6的执行时间的结束被上界定于灵活性间隔t_7的结束,而不管执行间隔t_6的限定的结束如何,例如,如果需要,从而允许所述任务sw_6比其分配的执行间隔更长地执行。

[0254] 情况2:

[0255] 如上文详细地描述的,包括三个任务sw_7、sw_8和sw_9的又一示例性任务集即第二任务集TS_2在表5中被表征。包括本发明的优点的时间触发调度器可以计算如表6中表征的具有用于任务的灵活性间隔的时间触发的调度,如图3中所描绘的。在该示例中,灵活性间隔相等。

[0256] 出于该示例的目的,图7描绘了所述任务集的可能执行轨迹,其中,任务集中的一些任务的任务执行时间大于对应的任务执行间隔。

[0257] 根据基于本发明的时间触发调派器,每个任务的开始执行时间由通过时间触发的调度确定的对应灵活性间隔界定,其中,所述灵活性间隔对于任务集中的所有任务是相等的。例如,在示例性轨迹中,图7中的sw_9的执行在其执行间隔t_5的开始之前的时间t_0处并且在具有较早执行间隔的任务sw_7和sw_8之前开始,例如这是由于任务sw_9具有比其预算执行间隔s_9更短的执行时间,该预算执行间隔s_9例如通过对先前循环的历史观察来识别,或者由于基于静态限定或动态分配的优先级来调派任务,由此sw_9被分配了比sw_8和sw_7更高的优先级。

[0258] 此外,在根据图7的示例中,任务sw_8的执行时间(t_7-t_3)大于与其执行间隔t_4-t_3对应的执行预算,从而允许sw_8的更大执行时间,该更大执行时间例如包括与来自先前任务的未使用时间加上灵活性间隔内的附加时间的累积松弛,从而例如允许sw_8实现使其执行时间最大化的渐进算法,直到灵活性间隔结束,由此所述算法的功能性能与其运行时成比例地改进。

[0259] 情况3:

[0260] 如上文详细描述的,包括三个任务sw_{10}、sw_{11}和sw_{12}的另一示例性任务TS_3即第三任务集在表7中被表征。时间触发调度器可以通过执行间隔的开始o、执行间隔的长度b、相关任务sw和每个执行间隔的灵活性间隔fi来计算如表8中描述的时间触发的调度,并且在图4中被描绘。用于第三任务集的所述时间触发的调度被提供有根据本发明的实施方式计算的灵活性间隔,并且其中,所述灵活性间隔部分地交叠。

[0261] 出于该示例的目的,图8描绘了所述第三任务集的可能执行轨迹,其中,任务集中的一些任务的任务执行时间大于对应的任务执行间隔。

[0262] 根据基于本发明的时间触发调派器,每个任务的开始执行时间由通过时间触发的调度确定的对应灵活性间隔界定,这些灵活性间隔部分地交叠。例如,在示例性轨迹中,图8中的sw_{10}的执行可以在其执行间隔t_2的开始之前的时间t_0处开始。类似地,任务sw_{11}可以就在sw_{10}完成执行之后并且在其执行间隔t_4的开始之前在时间t_{21}处开始。

[0263] 此外,图8中的任务sw_{12}的执行时间(t_{61}-t_{41})大于与其执行间隔t_7-t_6对应的执行预算,从而允许用于sw_{12}的更大的执行时间。

[0264] 参考文献

[0265] [1] S.S. Craciunas、R. Serna Oliver的Combined Task- and Network-level Scheduling for Distributed Time-triggered Networked Systems。实时系统,第52卷,第2期,第161至200页,施普林格(Springer),2016年。

[0266] [2] Anna Minaeva和Zdeněk Hanzálek。2021年。Survey on Periodic Scheduling for Time-triggered Hard Real-time Systems。ACM 计算机调查,第54卷,第1期,第23篇文章(2022年1月),32页。https://doi.org/10.1145/3431232

[0267] [3] Richard Hladik、Anna Minaeva和Zdeněk Hanzálek。2020年。On the Complexity of a Periodic Scheduling Problem with Precedence Relations。组合优化及其应用:第14届国际会议,COCOA 2020论文集,美国德克萨斯州达拉斯市,2020年12月11-13日。施普林格出版社,柏林,海德堡,第107至124页。https://doi.org/10.1007/978-3-030-64843-5_8

[0268] [4] McLean SD、Juul Hansen EA、Pop P和Craciunas SS(2022年)的Configuring ADAS Platforms for Automotive Applications Using Metaheuristics.Front。机器人与人工智能前沿。第8卷:762227。doi: 10.3389/frobt.2021.762227

[0269] [5] A. F. Mills和J. H. Anderson的“A Multiprocessor Server-Based Scheduler for Soft Real-Time Tasks with Stochastic Execution Demand”,2011年第17届IEEE嵌入式和实时计算系统及应用国际会议,日本富山市,2011年,第207至217页,doi:10.1109/RTCSA.2011.30.

[0270] [6] Ramon Serna Oliver、Paraskevas Karachatzis和Silviu Craciunas的

“Method to execute a mode-transition in a multi-mode computer system.”美国专利申请第17/930,811号。

[0271] [7] Baruah,S.K.、Rosier,L.E.和Howell, R.R.的Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor.实时系统2,第301-324页(1990年)。https://doi.org/10.1007/BF01995675

[0272] [8] R.Pellizzoni和G. Lipari的“Feasibility analysis of real-time periodic tasks with offsets”,实时系统,第30卷,第1-2期,第105-128页,2005年

[0273] [9] Kong, W;Nabi, M.;Goossens, K. Run的Time Recovery and Failure Analysis of Time-Triggered Traffic in Time Sensitive Networks. IEEE Access。2021年,第9卷,第91710-91722页。

[0274] [10] Li, J.;Xiong, H.;Li, Q.;Xiong, F.;Feng, J. Run的Time Reconfiguration Strategy and Implementation of Time-Triggered Networks。电子学,2022年,第11卷,1477。https://doi.org/10.3390/ electronics11091477。

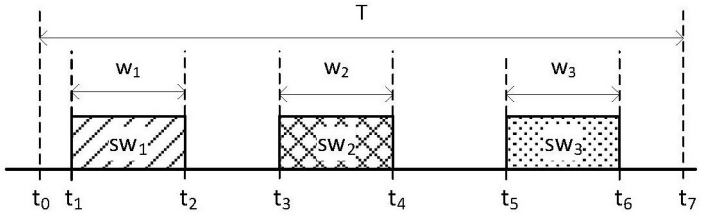

T
w1
w2
w3
SW1
SW2
SW3
t0 t1 t2 t3 t4 t5 t6 t7

图 1

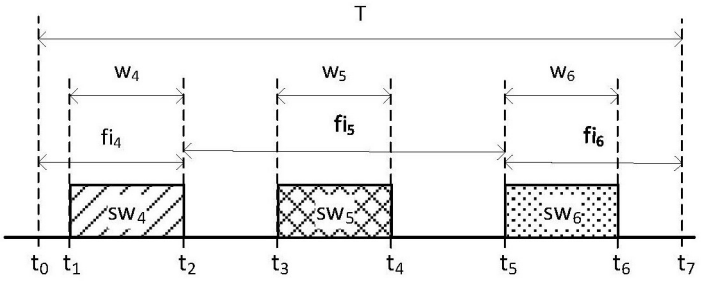

T
w4
w5
w6
fi4
fi5
fi6
SW4
SW5
SW6
t0 t1 t2 t3 t4 t5 t6 t7

图 2

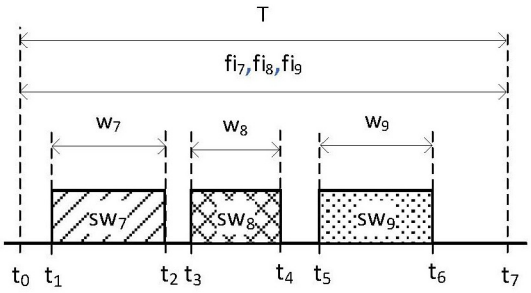

T
fi7,fi8,fi9
w7
w8
w9
SW7
SW8
SW9
t0 t1 t2 t3 t4 t5 t6 t7

图 3

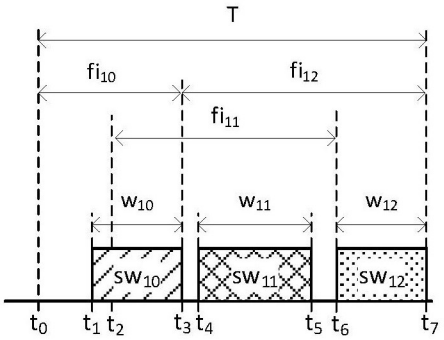

T
fi10
fi12
fi11
w10
w11
w12
SW10
SW11
SW12
t0 t1 t2 t3 t4 t5 t6 t7

图 4

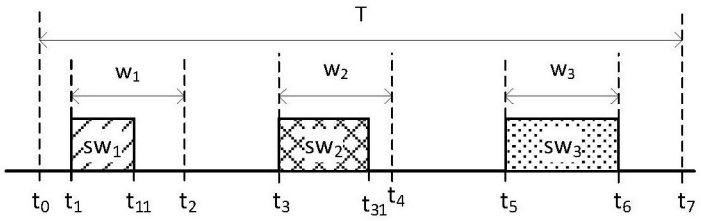

T
w1
w2
w3
SW1
SW2
SW3
t0 t1 t11 t2 t3 t31 t4 t5 t6 t7

图 5

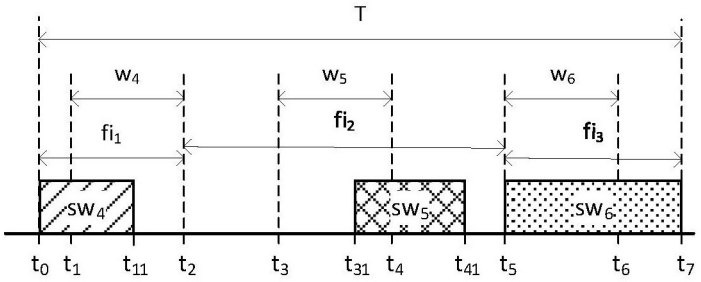

T
w4
w5
w6
fi1
fi2
fi3
SW4
SW5
SW6
t0 t1 t11 t2 t3 t31 t4 t41 t5 t6 t7

图 6

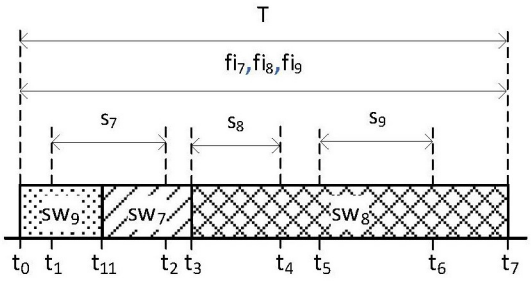

T
fi7,fi8,fi9
s7
s8
s9
SW9
SW7
SW8
t0 t1 t11 t2 t3 t4 t5 t6 t7

图 7

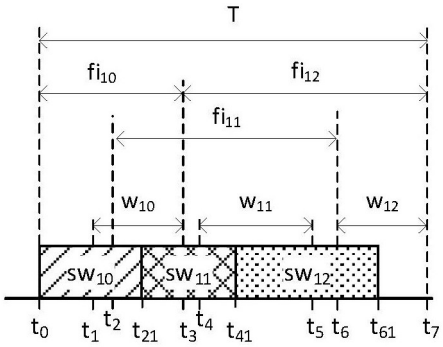

T
fi10
fi12
fi11
w10
w11
w12
SW10
SW11
SW12
t0 t1 t2 t21 t3 t4 t41 t5 t6 t61 t7

图 8