(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 119166301 B
(45)授权公告日 2025.02.18

(21)申请号 202411200763.0
(22)申请日 2024.08.29
(65)同一申请的已公布的文献号
申请公布号 CN 119166301 A
(43)申请公布日 2024.12.20
(73)专利权人 上海市大数据中心
地址 200040 上海市静安区寿阳路99弄15号
(72)发明人 张向飞 马燕萍 龚玎琦 朱策 梁永茂 袁大为 沈天杰
(74)专利代理机构 上海邦德专利代理事务所(普通合伙) 31312
专利代理师 杨益
(51)Int.Cl.
G06F 9/48(2006.01)
G06F 9/52(2006.01)
G06F 11/30(2006.01)
(56)对比文件
CN 102184125 A,2011.09.14
CN 110532247 A,2019.12.03
审查员 李若童

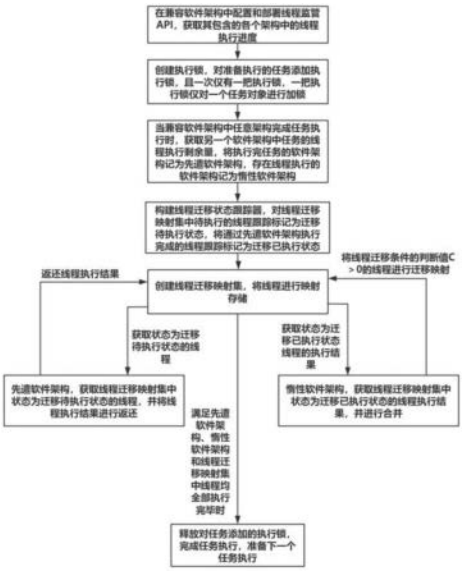
权利要求书3页 说明书7页 附图2页

(54)发明名称

一种兼容软件架构的应用分析监管系统及方法

(57)摘要

本发明公开了一种兼容软件架构的应用分析监管系统及方法,涉及兼容软件架构应用技术领域,本发明通过配置和部署线程监管API和创建执行锁,实时监控不同软件架构中的任务执行进度,确保任务在多架构环境中的同步执行,避免资源竞争和调度失,对线程在不同软件架构间的迁移情况进行判断和管理,确保任务执行的高效性和准确性,降低系统集成复杂性,保证任务执行的同步性和数据一致性,在多架构环境中进行线程迁移判断和状态跟踪,有效提高了系统的执行精确度,避免资源不足或软件架构性能差异大的情况下造成的资源浪费,确保任务在不同软件架构中的协调性,支持对不同软件架构的扩展,提高了系统的灵活性和适应性,适应多元化的计算平台需求。



CN 119166301 B

1.一种兼容软件架构的应用分析监管方法,其特征在于:所述应用分析监管方法包括以下步骤:

步骤S1、配置和部署兼容软件架构中的线程监管API,实时监管兼容软件架构中各个软件架构对应任务的线程执行进度;

步骤S2、创建执行锁,并在任务准备执行时为其添加执行锁,且兼容软件架构仅对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,获取另一软件架构的线程执行剩余量,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行的软件架构记为惰性软件架构;

创建执行锁,使用兼容软件架构对添加有执行锁的任务执行步骤如下:

步骤S2-1、创建执行锁,任务准备执行时,对任务添加执行锁,使执行锁与任务绑定,且每次仅对一个任务添加执行锁;

步骤S2-2、使用兼容软件架构对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,通过线程监管API获取另一软件架构的线程执行剩余量;

步骤S2-3、在兼容软件架构中,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行剩余量的软件架构记为惰性软件架构;

步骤S3、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,将满足线程迁移条件的线程映射到线程迁移映射集中;

步骤S4、构建线程迁移状态跟踪器,使用线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记;

步骤S5、调用先遣软件架构读取线程迁移映射集中对应线程迁移状态的线程,并将获取到的线程进行迁移执行,线程执行完成后,将线程执行结果返还到线程迁移映射集中,并使用线程迁移状态跟踪器对线程状态进行跟踪标记;

步骤S6、调用惰性软件架构将线程迁移映射集中已执行完毕的线程进行读取合并,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,释放步骤S2对任务添加的执行锁,执行下一个任务。

2.根据权利要求1所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述步骤S1中,对兼容软件架构中各个软件架构对应的任务线程执行进度监管步骤如下:

步骤S1-1、配置线程监管API的监控参数,并将线程监管API集成至兼容软件架构的各个模块中;

步骤S1-2、将配置完成的线程监管API部署到兼容软件架构中,使用线程监管API实时监管各个软件架构中任务的线程执行进度。

3.根据权利要求2所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述步骤S3的具体步骤如下:

步骤S3-1、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,迁移条件判断计算公式如下:

$$C=\left(\frac{T_{sent}-T_{fall}}{T_{all}}\right)+K*\left(\frac{V_{sent}}{V_{fall}}\right)-L*U_{flit};$$

式中,C表示为线程迁移条件的判断值;T_{sent}表示为先遣软件架构上的线程执行时间;T_{fall}表示为惰性软件架构上的线程执行时间;T_{all}表示为任务没有受到资源限制或架构差异的情况下,完成任务所需的时间;K和L均表示为权重因子;U_{flit}表示为线程迁移执行需要的线程迁移时间;

步骤S3-2、将线程迁移判断条件值C＞0的线程映射到线程迁移映射集中,线程迁移判断条件值C≤0的线程不做迁移映射,由惰性软件架构对其进行执行。

4.根据权利要求3所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述步骤S4中,构建线程迁移状态跟踪器,通过线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记,将线程迁移映射集中待执行的线程跟踪标记为迁移待执行状态,将执行完毕的线程跟踪标记为迁移已执行状态。

5.根据权利要求4所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述步骤S5中,先遣软件架构读取并执行线程的具体步骤如下:

步骤S5-1、调用先遣软件架构读取线程迁移映射集中状态为迁移待执行状态的线程,并对其线程进行执行,线程执行完毕后,将线程执行结果返还到线程迁移映射集中;

步骤S5-2、使用线程迁移状态跟踪器,将由先遣软件架构执行完毕的线程将状态由迁移待执行状态更改为迁移已执行状态。

6.根据权利要求5所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述步骤S6的具体步骤如下:

步骤S6-1、调用惰性软件架构读取线程迁移映射集中状态为迁移已执行状态的线程执行结果,将线程执行结果与自身架构中对应的数据进行校验合并;

步骤S6-2、当惰性软件架构读取完线程迁移映射集中的线程执行结果以后,销毁线程迁移映射集中状态为迁移已执行的线程,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,任务执行完毕,释放线程占有资源和步骤S2对任务添加的执行锁,执行下一个任务。

7.一种兼容软件架构的应用分析监管系统,其应用于权利要求1-6任意一项所述的一种兼容软件架构的应用分析监管方法,其特征在于:所述应用分析监管系统包括线程监管模块和执行锁模块;

所述线程监管模块包括线程日志记录单元、线程资源调度单元、线程参数配置单元和线程执行进度单元;所述线程日志记录单元用于记录线程的执行日志,线程执行日志包括线程执行的状态变化、性能数据和异常情况;所述线程资源调度单元用于管理线程使用的内存资源、文件句柄资源、网络连接资源,以及调度线程的执行顺序,基于线程的优先级、依赖关系和资源可用性来分配CPU时间片;所述线程参数配置单元用于设置和调整线程的执行参数;所述线程执行进度单元用于实时获取并记录线程执行的进度;

所述执行锁模块包括任务锁定单元、执行锁释放单元和超时处理单元;所述任务锁定单元用于对准备执行的任务进行加锁,通过加锁控制后续任务不可以插队执行,使兼容软件架构一次仅对一个任务进行执行,控制任务在不同软件架构中的执行进度;所述执行锁释放单元用于对兼容软件架构执行完任务后,对任务进行解锁,释放资源;所述超时处理单元用于处理执行锁对任务锁定超时的情况,当某个任务在规定时间内没有释放锁,超时处

理单元会自动解锁,采取相应措施来防止系统资源被长时间占用。

8.根据权利要求7所述的一种兼容软件架构的应用分析监管系统,其特征在于:所述应用分析监管系统还包括线程迁移判断模块和线程迁移映射模块;

所述线程迁移判断模块包括性能评估单元、线程迁移条件判断单元和线程资源匹配单元;所述性能评估单元用于评估当前线程在不同架构上的执行效率,通过比较线程在源架构和目标架构上的执行性能;所述线程迁移条件判断单元用于根据线程执行时间、资源使用率和架构性能差异判断线程是否满足迁移条件;所述线程资源匹配单元用于评估目标架构的资源情况,确保目标架构有足够的计算资源和内存空间来执行迁移过来的线程;

所述线程迁移映射模块包括线程映射单元和线程映射存储单元;所述线程映射单元用于将线程从惰性软件架构中映射到线程迁移映射集中;所述线程映射存储单元用于存储由线程映射单元映射出的线程。

9.根据权利要求7所述的一种兼容软件架构的应用分析监管系统,其特征在于:所述应用分析监管系统还包括线程状态跟踪标记模块和线程执行合并模块;

所述线程状态跟踪标记模块包括线程状态采集单元、状态标记单元和状态查询单元;所述线程状态采集单元用于实时采集迁移映射集中线程的执行状态;所述状态标记单元用于根据线程执行情况,对线程迁移映射集中的线程进行状态标记和更新;所述状态查询单元用于提供查询接口,通过查询接口查询线程状态的标记信息;

所述线程执行合并模块包括线程执行结果采集单元、结果合并单元和结果返还单元;所述执行结果采集单元用于采集各个线程在不同软件架构中执行后的结果数据;所述结果合并单元用于将不同线程的执行结果进行合并处理,形成完整的任务执行结果;所述结果返还单元用于将各个线程在不同软件架构中执行的结果进行集中返还到线程迁移映射集里。

一种兼容软件架构的应用分析监管系统及方法

技术领域

[0001] 本发明涉及兼容软件架构应用技术领域,具体是一种兼容软件架构的应用分析监管系统及方法。

背景技术

[0002] 随着计算平台的多样化发展,兼容不同架构(如ARM和x86)在软件系统中的重要性日益增加。ARM架构以其高能效和低成本广泛应用于移动设备和嵌入式系统,而x86架构则在高性能计算和桌面应用中占据主导地位。兼容不同架构的软件系统能够在不同硬件平台上运行,确保应用的广泛适用性,降低开发和维护成本,同时提高系统的灵活性和扩展性。在当今多元化的计算环境中,这种兼容性有助于企业应对不同市场需求,保持竞争优势,尤其是在信创领域,更为关键。

[0003] 在兼容不同架构的软件系统中,由于ARM和x86架构的指令集和执行效率差异,相同任务在不同架构上会出现进度不一致的问题。这种不一致性可能导致任务调度失衡,影响整体系统性能,尤其在实时性要求高的应用中尤为明显。此外,进度不一致还会引发数据同步困难和资源浪费,增加系统复杂度和维护成本。因此,解决这一问题对于确保任务在多架构环境中平稳运行至关重要。

发明内容

[0004] 本发明的目的在于提供一种兼容软件架构的应用分析监管系统及方法,以解决现有技术中提出的的问题。

[0005] 为实现上述目的,本发明提供如下技术方案:一种兼容软件架构的应用分析监管方法,应用分析监管方法包括以下步骤:

[0006] 步骤S1、配置和部署兼容软件架构中的线程监管API,实时监管兼容软件架构中各个软件架构对应任务的线程执行进度;

[0007] 步骤S1-1、配置线程监管API的监控参数,并将线程监管API集成至兼容软件架构的各个模块中;

[0008] 步骤S1-2、将配置完成的线程监管API部署到兼容软件架构中,使用线程监管API实时监管各个软件架构中任务的线程执行进度;

[0009] 通过配置线程监管API的监控参数,能够更好地了解不同架构中的线程执行情况,集成线程监管API至各个模块中,确保各模块之间的任务执行进度同步。这样可以减少不同架构之间的进度不一致问题,确保任务在不同架构中能够协调一致地执行,还可以简化监控开发和调试操作,减少开发和集成的复杂性。

[0010] 步骤S2、创建执行锁,并在任务准备执行时为其添加执行锁,且兼容软件架构仅对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,获取另一软件架构的线程执行剩余量,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行的软件架构记为惰性软件架构;

[0011] 步骤S2-1、创建执行锁,任务准备执行时,对任务添加执行锁,使执行锁与任务绑定,且每次仅对一个任务添加执行锁;

[0012] 步骤S2-2、使用兼容软件架构对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,通过线程监管API获取另一软件架构的线程执行剩余量;

[0013] 步骤S2-3、在兼容软件架构中,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行剩余量的软件架构记为惰性软件架构;

[0014] 创建的执行锁,一次仅对一个任务进行添加,兼容软件架构执行任务时,只对添加有执行锁的任务进行读取并执行,可以保证任务在不同软件架构中的执行进度一致,解决数据同步困难的问题。

[0015] 步骤S3、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,将满足线程迁移条件的线程映射到线程迁移映射集中;

[0016] 步骤S3-1、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,迁移条件判断计算公式如下:

[0017] $$C=\left(\frac{T_{sent}-T_{fall}}{T_{all}}\right)+K*\left(\frac{V_{sent}}{V_{fall}}\right)-L*U_{flit};$$

[0018] 式中,C表示为线程迁移条件的判断值;T_{sent}表示为先遣软件架构上的线程执行时间;T_{fall}表示为惰性软件架构上的线程执行时间;T_{all}表示为任务没有受到资源限制或架构差异的情况下,完成任务所需的时间;K和L均表示为权重因子;U_{flit}表示为线程迁移执行需要的线程迁移时间;

[0019] 步骤S3-2、将线程迁移判断条件值C>0的线程映射到线程迁移映射集中,线程迁移判断条件值C≤0的线程不做迁移映射,由惰性软件架构对其进行执行;

[0020] 通过对惰性软件架构上待执行的线程进行迁移条件计算判断,避免线程迁移执行后的成本大于在惰性软件架构上执行的成本,高效的提升了线程迁移执行的效率,避免资源浪费和任务执行过长的问题。

[0021] 步骤S4、构建线程迁移状态跟踪器,使用线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记;

[0022] 步骤S5、调用先遣软件架构读取线程迁移映射集中对应线程迁移状态的线程,并将获取到的线程进行迁移执行,线程执行完成后,将线程执行结果返还到线程迁移映射集中,并使用线程迁移状态跟踪器对线程状态进行跟踪标记;

[0023] 步骤S5-1、调用先遣软件架构读取线程迁移映射集中状态为迁移待执行状态的线程,并对其线程进行执行,线程执行完毕后,将线程执行结果返还到线程迁移映射集中;

[0024] 步骤S5-2、使用线程迁移状态跟踪器,将由先遣软件架构执行完毕的线程将状态由迁移待执行状态更改为迁移已执行状态;

[0025] 通过对线程迁移映射集中的线程进行状态标识,避免了惰性软件架构在对迁移映射集中进行线程结果获取时,获取到状态为迁移待执行状态的线程,或者避免先遣软件架构对迁移映射集中进行线程获取时,获取到状态为迁移已执行状态的线程,实现高效的线程获取,加快软件架构处理线程的速度与精度。

[0026] 步骤S6、调用惰性软件架构将线程迁移映射集中已执行完毕的线程进行读取合并,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,释放步骤S2对任务添加的执行锁,执行下一个任务。

[0027] 步骤S6-1、调用惰性软件架构读取线程迁移映射集中状态为迁移已执行状态的线程执行结果,将线程执行结果与自身架构中对应的数据进行校验合并;

[0028] 步骤S6-2、当惰性软件架构读取完线程迁移映射集中的线程执行结果以后,销毁线程迁移映射集中状态为迁移已执行的线程,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,任务执行完毕,释放线程占有资源和步骤S2对任务添加的执行锁,执行下一个任务;

[0029] 通过调用惰性软件架构读取并合并已执行完毕的线程,确保任务的执行结果准确无误。当所有线程完成执行后,系统自动销毁已执行的线程,释放占用的资源和任务执行锁。这种机制能够优化任务执行的准确性和效率,避免系统资源浪费,确保系统顺利进入下一个任务执行阶段,提升多架构兼容系统的整体性能和任务处理能力。

[0030] 进一步的,在步骤S4中,构建线程迁移状态跟踪器,通过线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记,将线程迁移映射集中待执行的线程跟踪标记为迁移待执行状态,将执行完毕的线程跟踪标记为迁移已执行状态。

[0031] 进一步的,一种兼容软件架构的应用分析监管系统包括线程监管模块、执行锁模块、线程迁移判断模块、线程迁移映射模块、线程状态跟踪标记模块和线程执行合并模块;

[0032] 所述线程监管模块包括线程日志记录单元、线程资源调度单元、线程参数配置单元和线程执行进度单元;所述线程日志记录单元用于记录线程的执行日志,线程执行日志包括线程执行的状态变化、性能数据和异常情况;所述线程资源调度单元用于管理线程使用的内存资源、文件句柄资源、网络连接资源,以及调度线程的执行顺序,基于线程的优先级、依赖关系和资源可用性来分配CPU时间片;所述线程参数配置单元用于设置和调整线程的执行参数;所述线程执行进度单元用于实时获取并记录线程执行的进度;

[0033] 系统内的每个线程都会被分配一个优先级,优先级由操作系统或应用程序根据线程的重要性和紧急性来设定,由系统执行线程的逻辑进行分配,系统会尽量将有依赖关系的线程安排在接近的时间片内,以减少等待时间和资源竞争,在分配CPU时间片时,系统不仅考虑线程的优先级,还要考虑当前可用的系统资源内存和I/O带宽。

[0034] 所述执行锁模块包括任务锁定单元、执行锁释放单元和超时处理单元;所述任务锁定单元用于对准备执行的任务进行加锁,通过加锁控制后续任务不可以插队执行,使兼容软件架构一次仅对一个任务进行执行,控制任务在不同软件架构中的执行进度;所述执行锁释放单元用于对兼容软件架构执行完任务后,对任务进行解锁,释放资源;所述超时处理单元用于处理执行锁对任务锁定超时的情况,当某个任务在规定时间内没有释放锁,超时处理单元会自动解锁,采取相应措施来防止系统资源被长时间占用;

[0035] 所述线程迁移判断模块包括性能评估单元、线程迁移条件判断单元和线程资源匹配单元;所述性能评估单元用于评估当前线程在不同架构上的执行效率,通过比较线程在源架构和目标架构上的执行性能;所述线程迁移条件判断单元用于根据线程执行时间、资源使用率和架构性能差异判断线程是否满足迁移条件;所述线程资源匹配单元用于评估目标架构的资源情况,确保目标架构有足够的计算资源和内存空间来执行迁移过来的线程;

[0036] 所述线程迁移映射模块包括线程映射单元和线程映射存储单元;所述线程映射单元用于将线程从惰性软件架构中映射到线程迁移映射集中;所述线程映射存储单元用于存储由线程映射单元映射出的线程;

[0037] 所述线程状态跟踪标记模块包括线程状态采集单元、状态标记单元和状态查询单元;所述线程状态采集单元用于实时采集迁移映射集中线程的执行状态;所述状态标记单元用于根据线程执行情况,对线程迁移映射集中的线程进行状态标记和更新;所述状态查询单元用于提供查询接口,通过查询接口查询线程状态的标记信息;

[0038] 所述线程执行合并模块包括线程执行结果采集单元、结果合并单元和结果返还单元;所述执行结果采集单元用于采集各个线程在不同软件架构中执行后的结果数据;所述结果合并单元用于将不同线程的执行结果进行合并处理,形成完整的任务执行结果;所述结果返还单元用于将各个线程在不同软件架构中执行的结果进行集中返还到线程迁移映射集里。

[0039] 与现有技术相比,本发明的有益效果是:本发明通过配置和部署线程监管API和创建执行锁,实时监控不同架构中的任务执行进度,确保任务在多架构环境中的同步执行,避免资源竞争和调度失衡;通过创建线程迁移映射集和使用状态跟踪器,对线程在不同架构间的迁移情况进行判断和管理,确保任务执行的高效性和准确性;

[0040] 1、本发明通过配置和部署线程监管API,实时监控线程的执行进度,并集成至各个模块中,简化了开发和调试过程,降低了系统集成的复杂性,确保了任务执行的同步性和数据一致性。

[0041] 2、本发明通过在多架构环境中进行线程迁移和状态跟踪,有效提高了系统的执行效率,避免了在资源不足或架构性能差异大的情况下造成的资源浪费,确保了任务在不同架构中的协调一致性。

[0042] 3、本发明通过设置执行锁,以及线程迁移判断和线程状态跟踪标记的设计,避免了线程在多架构环境中的执行混乱和进度不一致问题,提升了系统的稳定性。同时,支持对不同架构的扩展,提高了系统的灵活性和适应性,适应多元化的计算平台需求。

附图说明

[0043] 图1为本发明一种兼容软件架构的应用分析监管方法的流程示意图;

[0044] 图2为本发明一种兼容软件架构的应用分析监管系统的结构示意图。

具体实施方式

[0045] 下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅仅是本发明一部分实施例,而不是全部的实施例。基于本发明中的实施例,本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例,都属于本发明保护的范围。

[0046] 实施例一:如图1所示,本发明提供一种技术方案,一种兼容软件架构的应用分析监管方法,应用分析监管方法包括以下步骤:

[0047] 步骤S1、配置和部署兼容软件架构中的线程监管API,实时监管兼容软件架构中各个软件架构对应任务的线程执行进度;

[0048] 步骤S1-1、配置线程监管API的监控参数,并将线程监管API集成至兼容软件架构的各个模块中;

[0049] 步骤S1-2、将配置完成的线程监管API部署到兼容软件架构中,使用线程监管API实时监管各个软件架构中任务的线程执行进度;

[0050] 例如,视频处理任务分为两个部分:视频解码(在ARM上执行)和视频编码(在x86上执行)。ARM上的解码任务需要10秒,x86上的编码任务需要15秒;

[0051] 配置和部署线程监管API,ARM架构的线程ID为101,x86架构的线程ID为201;

[0052] 第1秒:ARM线程101的CPU使用率为30%,执行时间为1秒;x86线程202的CPU使用率为50%,执行时间为1秒。

[0053] 第5秒:ARM线程101的状态为运行中,执行时间为5秒;x86线程202的状态为运行中,执行时间为5秒。

[0054] 第10秒:ARM线程101完成,x86线程202还在执行。

[0055] 步骤S2、创建执行锁,并在任务准备执行时为其添加执行锁,且兼容软件架构仅对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,获取另一软件架构的线程执行剩余量,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行的软件架构记为惰性软件架构;

[0056] 步骤S2-1、创建执行锁,任务准备执行时,对任务添加执行锁,使执行锁与任务绑定,且每次仅对一个任务添加执行锁;

[0057] 步骤S2-2、使用兼容软件架构对添加有执行锁的任务进行执行,当任务在兼容软件架构中的其中一种架构上执行完毕后,通过线程监管API获取另一软件架构的线程执行剩余量;

[0058] 步骤S2-3、在兼容软件架构中,将任务执行完毕的软件架构记为先遣软件架构,存在线程执行剩余量的软件架构记为惰性软件架构;

[0059] 步骤S3、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,将满足线程迁移条件的线程映射到线程迁移映射集中;

[0060] 步骤S3-1、创建线程迁移映射集,对惰性软件架构中待执行的线程进行线程迁移条件判断,迁移条件判断计算公式如下:

[0061] $$C=\left(\frac{T_{sent}-T_{fall}}{T_{all}}\right)+K*\left(\frac{V_{sent}}{V_{fall}}\right)-L*U_{flit};$$

[0062] 式中,C表示为线程迁移条件的判断值;T_{sent}表示为先遣软件架构上的线程执行时间;T_{fall}表示为惰性软件架构上的线程执行时间;T_{all}表示为任务没有受到资源限制或架构差异的情况下,完成任务所需的时间;K和L均表示为权重因子;U_{flit}表示为线程迁移执行需要的线程迁移时间,如数据复制,线程上下文切换等;

[0063] 例如,系统为,ARM架构,资源利用率高,x86架构,执行速度较快,ARM架构系统由于架构特性,线程执行时间较长,而x86架构系统执行同一线程的时间较短;

[0064] 设定条件:某线程在ARM架构系统上的执行时间为 T_{sent} = 30ms,在x86架构系统上的执行时间为 T_{fall} = 10ms,任务没有受到资源限制或架构差异情况下的完成时间 T_{all} =

50ms,设权重因子K为0.7,L为0.3,线程迁移成本U_{flit}为5ms,使用迁移条件判断计算公式计算:

[0065] C=0.7*(30-10)+0.3*(50-10)-5=21ms

[0066] 根据条件进行判断,满足C＞0,则将ARM架构系统上线程进行迁移映射到x86架构系统上执行。

[0067] 步骤S4、构建线程迁移状态跟踪器,使用线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记;

[0068] 步骤S5、调用先遣软件架构读取线程迁移映射集中对应线程迁移状态的线程,并将获取到的线程进行迁移执行,线程执行完成后,将线程执行结果返还到线程迁移映射集中,并使用线程迁移状态跟踪器对线程状态进行跟踪标记;

[0069] 步骤S5-1、调用先遣软件架构读取线程迁移映射集中状态为迁移待执行状态的线程,并对其线程进行执行,线程执行完毕后,将线程执行结果返还到线程迁移映射集中;

[0070] 步骤S5-2、使用线程迁移状态跟踪器,将由先遣软件架构执行完毕的线程将状态由迁移待执行状态更改为迁移已执行状态;

[0071] 步骤S6、调用惰性软件架构将线程迁移映射集中已执行完毕的线程进行读取合并,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,释放步骤S2对任务添加的执行锁,执行下一个任务。

[0072] 步骤S6-1、调用惰性软件架构读取线程迁移映射集中状态为迁移已执行状态的线程执行结果,将线程执行结果与自身架构中对应的数据进行校验合并;

[0073] 步骤S6-2、当惰性软件架构读取完线程迁移映射集中的线程执行结果以后,销毁线程迁移映射集中状态为迁移已执行的线程,当线程迁移映射集、先遣软件架构和惰性软件架构中的线程均执行完毕时,任务执行完毕,释放线程占有资源和步骤S2对任务添加的执行锁,执行下一个任务;

[0074] 例如,现有任务A和任务B,任务A排序在任务B之前,任务A在兼容ARM架构系统和x86架构系统的兼容软件架构中执行,任务A在ARM架构系统中执行需要10ms,在x86架构系统中执行需要5ms,任务A在准备执行时,创建的执行锁会对其进行添加绑定,兼容软件架构运行时,获取带有执行锁的任务A,任务B不可与任务A争夺系统资源,x86架构系统执行完任务A时,对ARM架构系统中的A任务的进度进行读取,将满足线程迁移条件的线程进行迁移执行,执行完线程后,将线程执行结果进行返还,判断ARM架构系统、线程迁移映射集和x86架构系统中的线程进行判断,当线程均执行完毕后,释放对任务A添加的执行锁,释放系统资源,准备下个任务的执行。

[0075] 构建线程迁移状态跟踪器,通过线程迁移状态跟踪器对线程迁移映射集中的线程迁移状态进行跟踪标记,将线程迁移映射集中待执行的线程跟踪标记为迁移待执行状态,将执行完毕的线程跟踪标记为迁移已执行状态。

[0076] 实施例二,如图2所示,本发明提供一种兼容软件架构的应用分析监管系统,应用分析监管系统包括线程监管模块、执行锁模块、线程迁移判断模块、线程迁移映射模块、线程状态跟踪标记模块和线程执行合并模块;

[0077] 所述线程监管模块包括线程日志记录单元、线程资源调度单元、线程参数配置单元和线程执行进度单元;所述线程日志记录单元用于记录线程的执行日志,线程执行日志

包括线程执行的状态变化、性能数据和异常情况;所述线程资源调度单元用于管理线程使用的内存资源、文件句柄资源、网络连接资源,以及调度线程的执行顺序,基于线程的优先级、依赖关系和资源可用性来分配CPU时间片;所述线程参数配置单元用于设置和调整线程的执行参数;所述线程执行进度单元用于实时获取并记录线程执行的进度;

[0078] 所述执行锁模块包括任务锁定单元、执行锁释放单元和超时处理单元;所述任务锁定单元用于对准备执行的任务进行加锁,通过加锁控制后续任务不可以插队执行,使兼容软件架构一次仅对一个任务进行执行,控制任务在不同软件架构中的执行进度;所述执行锁释放单元用于对兼容软件架构执行完任务后,对任务进行解锁,释放资源;所述超时处理单元用于处理执行锁对任务锁定超时的情况,当某个任务在规定时间内没有释放锁,超时处理单元会自动解锁,采取相应措施来防止系统资源被长时间占用;

[0079] 所述线程迁移判断模块包括性能评估单元、线程迁移条件判断单元和线程资源匹配单元;所述性能评估单元用于评估当前线程在不同架构上的执行效率,通过比较线程在源架构和目标架构上的执行性能;所述线程迁移条件判断单元用于根据线程执行时间、资源使用率和架构性能差异判断线程是否满足迁移条件;所述线程资源匹配单元用于评估目标架构的资源情况,确保目标架构有足够的计算资源和内存空间来执行迁移过来的线程;

[0080] 所述线程迁移映射模块包括线程映射单元和线程映射存储单元;所述线程映射单元用于将线程从惰性软件架构中映射到线程迁移映射集中;所述线程映射存储单元用于存储由线程映射单元映射出的线程;

[0081] 所述线程状态跟踪标记模块包括线程状态采集单元、状态标记单元和状态查询单元;所述线程状态采集单元用于实时采集迁移映射集中线程的执行状态;所述状态标记单元用于根据线程执行情况,对线程迁移映射集中的线程进行状态标记和更新;所述状态查询单元用于提供查询接口,通过查询接口查询线程状态的标记信息;

[0082] 所述线程执行合并模块包括线程执行结果采集单元、结果合并单元和结果返还单元;所述执行结果采集单元用于采集各个线程在不同软件架构中执行后的结果数据;所述结果合并单元用于将不同线程的执行结果进行合并处理,形成完整的任务执行结果;所述结果返还单元用于将各个线程在不同软件架构中执行的结果进行集中返还到线程迁移映射集里。

[0083] 对于本领域技术人员而言,显然本发明不限于上述示范性实施例的细节,而且在不背离本发明的精神或基本特征的情况下,能够以其他的具体形式实现本发明。因此,无论从哪一点来看,均应将实施例看作是示范性的,而且是非限制性的,本发明的范围由所附权利要求而不是上述说明限定,因此旨在将落在权利要求的等同要件的含义和范围内的所有变化囊括在本发明内。不应将权利要求中的任何附图标记视为限制所涉及的权利要求。

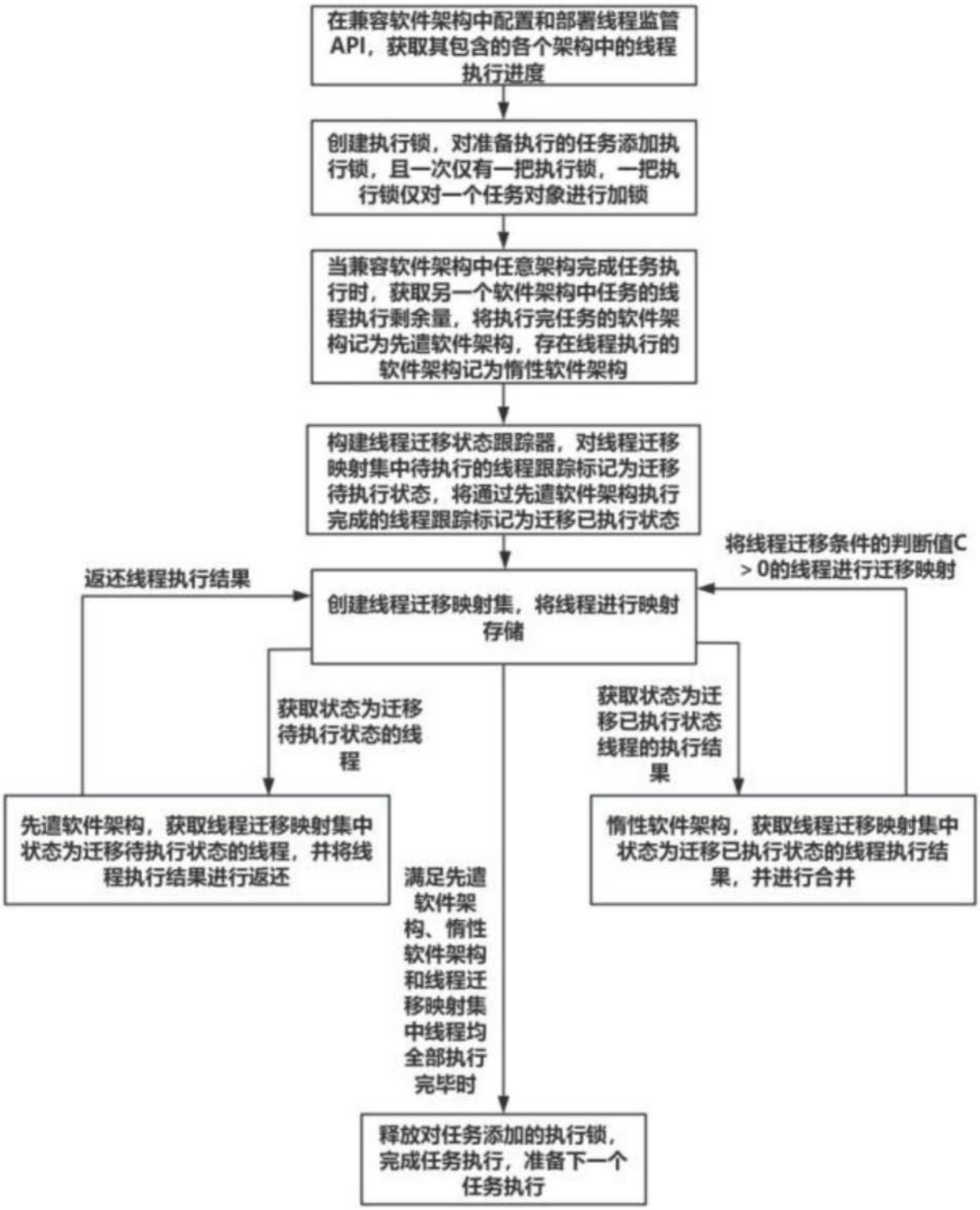

在兼容软件架构中配置和部署线程监管API，获取其包含的各个架构中的线程执行进度
创建执行锁，对准备执行的任务添加执行锁，且一次仅有一把执行锁，一把执行锁仅对一个任务对象进行加锁
当兼容软件架构中任意架构完成任务执行时，获取另一个软件架构中任务的线程执行剩余量，将执行完任务的软件架构记为先遣软件架构，存在线程执行的软件架构记为惰性软件架构
构建线程迁移状态跟踪器，对线程迁移映射集中待执行的线程跟踪标记为迁移待执行状态，将通过先遣软件架构执行完成的线程跟踪标记为迁移已执行状态
将线程迁移条件的判断值C>0的线程进行迁移映射
返还线程执行结果
创建线程迁移映射集，将线程进行映射存储
获取状态为迁移待执行状态的线程
获取状态为迁移已执行状态线程的执行结果
先遣软件架构，获取线程迁移映射集中状态为迁移待执行状态的线程，并将线程执行结果进行返还
惰性软件架构，获取线程迁移映射集中状态为迁移已执行状态的线程执行结果，并进行合并
满足先遣软件架构、惰性软件架构和线程迁移映射集中线程均全部执行完毕时
释放对任务添加的执行锁，完成任务执行，准备下一个任务执行

图1

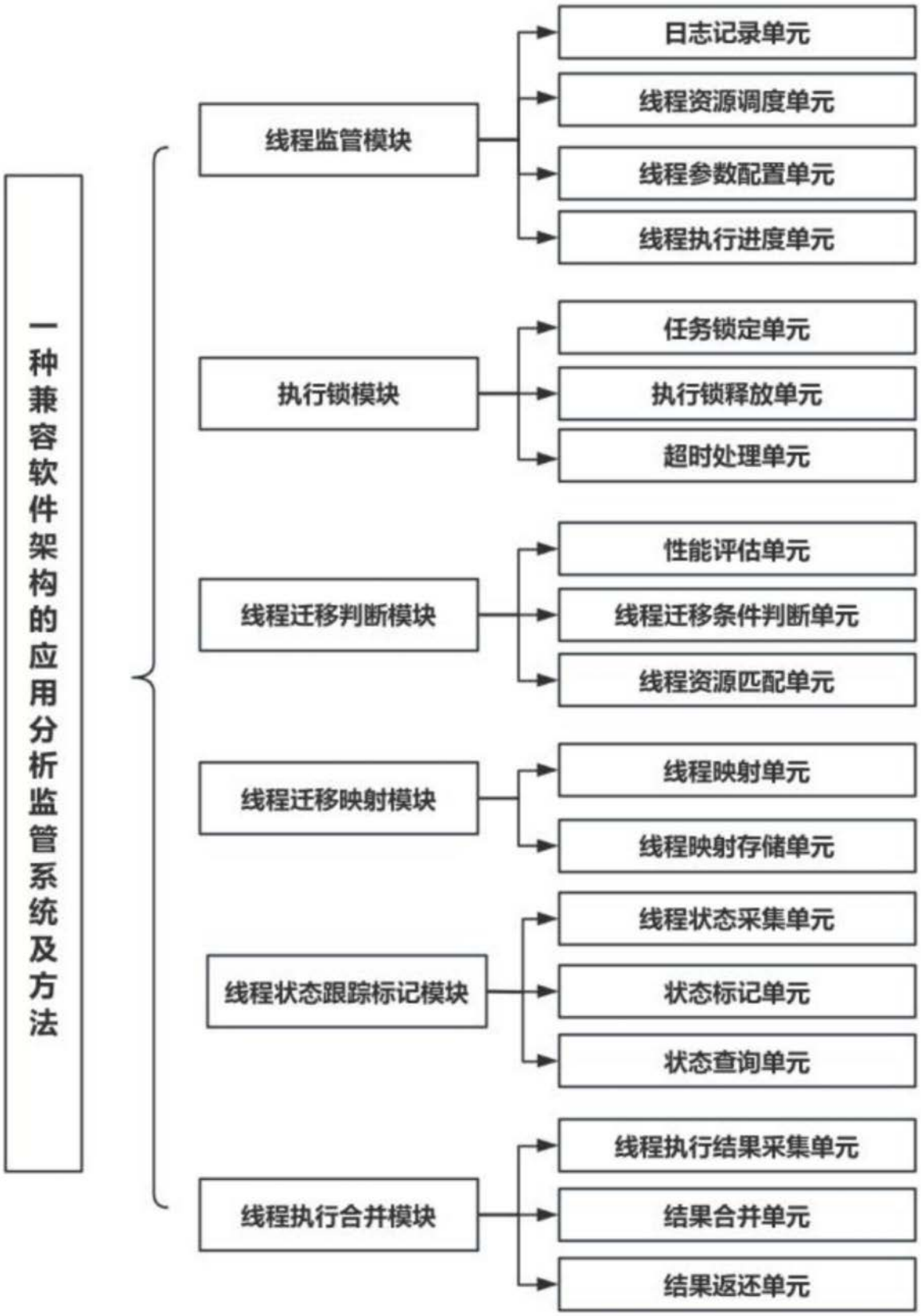

一种兼容软件架构的应用分析监管系统及方法
线程监管模块
日志记录单元
线程资源调度单元
线程参数配置单元
线程执行进度单元
执行锁模块
任务锁定单元
执行锁释放单元
超时处理单元
线程迁移判断模块
性能评估单元
线程迁移条件判断单元
线程资源匹配单元
线程迁移映射模块
线程映射单元
线程映射存储单元
线程状态跟踪标记模块
线程状态采集单元
状态标记单元
状态查询单元
线程执行合并模块
线程执行结果采集单元
结果合并单元
结果返还单元

图2