(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国 际 局

(43) 国际公布日
2023 年 11 月 23 日 (23.11.2023)

WIPO | PCT

(10) 国际公布号
WO 2023/224549 A2

(51) 国际专利分类号:
无分类

(21) 国际申请号: PCT/SG2023/050309

(22) 国际申请日: 2023 年 5 月 5 日 (05.05.2023)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
202210527828.7 2022年5月16日 (16.05.2022) CN

(71) 申请人:脸萌有限公司(**LEMON INC.**) [GB/SG]; 开曼群岛大开曼西湾路802号木槿道大楼邮政信箱31119, Grand Cayman KY1 - 1205 (KY)。

(72) 发明人:张乐林(**ZHANG, Lelin**); 美国加利福尼亚州洛杉矶西杰弗逊大道12655号六层第137室, California 90066 (US)。 杨池良(**YANG, Chiliang**); 中国北京市朝阳区七圣中街12号院融中心B1小邮局, Beijing 100028 (CN)。 蒋祖尧(**JIANG, Zuyao**); 中国北京市朝阳区七圣中街12号院融中心B1小邮局, Beijing 100028 (CN)。 梅星(**MEI, Xing**); 美国加利福尼亚州洛杉矶西杰弗逊大道12655号六层第137室, California 90066 (US)。

(74) 代理人:傅子健(**POH, Chee Kian, Daniel**); 新加坡新加坡麥仕奇新加坡,丹戎巴葛邮局,POBox636, Singapore 910816 (SG)。

(81) 指定国(除另有指明，要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ,

(54) **Title:** SCHEDULING METHOD AND APPARATUS, DEVICE AND STORAGE MEDIUM

(54) 发明名称: 调度方法、装置、设备及存储介质

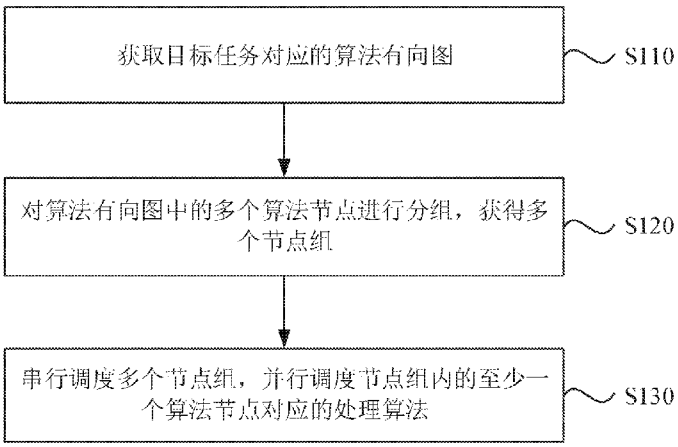

图 1

S110 Acquire an algorithm directed graph corresponding to a target task

S120 Group a plurality of algorithm nodes in the algorithm directed graph to obtain a plurality of node groups

S130 Serially schedule the plurality of node groups, and schedule in parallel a processing algorithm corresponding to at least one algorithm node in each node group

(57) **Abstract:** The embodiments of the present disclosure disclose a scheduling method and apparatus, a device and a storage medium. The method comprises: acquiring an algorithm directed graph corresponding to a target task, wherein the algorithm directed graph includes a plurality of algorithm nodes; grouping the plurality of algorithm nodes in the algorithm directed graph to obtain a plurality of node groups; and serially scheduling the plurality of node groups, and scheduling in parallel a processing algorithm corresponding to at least one algorithm node in each node group.

WO 2023/224549 A2

[见续页]

IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW。

(84) 指定国(除另有指明，要求每一种可提供的地区保护): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布:
— 不包括国际检索报告，在收到该报告后将重新公布(细则48.2(g))。

(57) 摘要: 说明书摘要本公开实施例公开了一种调度方法、装置、设备及存储介质。该方法包括：获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；对所述算法有向图中的多个算法节点进行分组，获得多个节点组；串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

调度方法、装置、设备及存储介质

本公开要求在2022年5月16日提交中国专利局、申请号为202210527828.7的中国专利申请的优先权，以上申请的全部内容通过引用结合在本公开中。

技术领域

本公开实施例涉及计算机技术领域，例如涉及一种调度方法、装置、设备及存储介质。

背景技术

随着计算机视觉和图形学的快速发展，通过人工智能（如：机器学习或者深度学习）和虚拟增强现实技术可以生成丰富多样的图像。

对于一个图像处理任务，通常需要调用多个算法实现。相关技术中，各算法以一定的顺序依次执行，影响了整个任务的吞吐率及计算效率。

发明内容

本公开实施例提供一种调度方法、装置、设备及存储介质，可以并行调度算法节点，提高目标任务处理时的吞吐率及计算效率。

第一方面，本公开实施例提供了一种调度方法，包括：

获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；

对所述算法有向图中的多个算法节点进行分组，获得多个节点组；

串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

第二方面，本公开实施例还提供了一种调度装置，包括：

算法有向图获取模块，设置为获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；

算法节点分组模块，设置为对所述算法有向图中的多个算法节点进行分组，获得多个节点组；

调度模块，设置为串行调度所述多个节点组，并行调度所述节点组内的至

少一个算法节点对应的处理算法。

第三方面，本公开实施例还提供了一种电子设备，所述电子设备包括：

一个或多个处理装置；

存储装置，设置为存储一个或多个程序；

当所述一个或多个程序被所述一个或多个处理装置执行，使得所述一个或多个处理装置实现如本公开实施例所述的调度方法。

第四方面，本公开实施例还提供了一种计算机可读介质，所述计算机可读介质上存储有计算机程序，所述计算机程序被处理装置执行时实现如本公开实施例所述的调度方法。

附图说明

图 1 是本公开实施例中的一种调度方法的流程图；

图 2 是本公开实施例中的一种算法有向图的示例图；

图 3 是本公开实施例中的一种算法有向图的示例图；

图 4 是本公开实施例中的对算法节点进行分组的示例图；

图 5 是本公开实施例中的对算法节点进行分组的示例图；

图 6 是本公开实施例中的算法调度示例图；

图 7 是本公开实施例中的一种调度装置的结构示意图；

图 8 是本公开实施例中的一种电子设备的结构示意图。

具体实施方式

下面将参照附图描述本公开的实施例。虽然附图中显示了本公开的某些实施例，然而应当理解的是，本公开可以通过各种形式来实现。应当理解的是，本公开的附图及实施例仅用于示例性作用。

应当理解，本公开的方法实施方式中记载的各个步骤可以按照不同的顺序执行，和/或并行执行。此外，方法实施方式可以包括附加的步骤和/或省略执行示出的步骤。

本文使用的术语“包括”及其变形是开放性包括，即“包括但不限于”。术语“基于”是“至少部分地基于”。术语“一个实施例”表示“至少一个实施例”；术语“另一实施例”表示“至少一个另外的实施例”；术语“一些实施例”表示“至少一些实施

例”。其他术语的相关定义将在下文描述中给出。

需要注意，本公开中提及的“第一”、“第二”等概念仅用于对不同的装置、模块或单元进行区分，并非用于限定这些装置、模块或单元所执行的功能的顺序或者相互依存关系。

需要注意，本公开中提及的“一个”、“多个”的修饰是示意性的，本领域技术人员应当理解，除非在上下文另有明确指出，否则应该理解为“一个或多个”。

本公开实施方式中的多个装置之间所交互的消息或者信息的名称仅用于说明性的目的，而并不是用于对这些消息或信息的范围进行限制。

智能创作：特指视频类社交平台中所使用的基于计算机视觉和图形学的图像视频内容生成方法，通过人工智能（例如传统机器学习或深度学习）和虚拟现实/增强现实技术的应用，使用户提供的视频内容具备更加多元丰富的内容。

算法平台：在用户使用移动平台或其他个人计算机（Personal Computer，PC）平台进行智能创作时所使用的支持算法调度与执行的软件系统，该系统的输入为来自相机的图片或视频信息以及所需要运行的算法，这些算法的执行顺序与依赖关系通过有向图来描述和连接，该系统的输出为算法运行的结果，包括图像分类信息、目标物体检测包围框及置信度、物体分割信息、生成图像、人体或物体关键点信息等。

并行异步框架：基于中央处理器（Central Processing Unit，CPU）多线程加速或其他硬件的软件优化框架，该框架的主要作用是在多线程技术的支持下，通过拆解和动态分配可以并发执行的任务来提高系统的整体吞吐率和计算效率。

在当前的智能创作计算平台上，对每一张图像，算法图中的各个计算节点均以顺序方式执行，且节点之间不存在多帧结果的缓存，造成了潜在的计算资源浪费。

图1为本公开实施例一提供的一种调度方法的流程图，本实施例可适用于并行调度算法图中的算法节点的情况，该方法可以由调度装置来执行，该装置可由硬件和/或软件组成，并一般可集成在具有调度方法功能的设备中，该设备可以是服务器、移动终端或服务器集群等电子设备。如图1所示，该方法可包括如下步骤：

S110，获取目标任务对应的算法有向图。

其中，算法有向图包含多个算法节点，各算法节点之间通过有向边连接，有向边两端的算法节点具有依赖关系，有向边结束端的算法节点依赖起始端的算法节点。示例性的，图2是本实施例中的一种算法有向图的示例图，如图2所示，该算法有向图包含6个算法节点，其中，算法节点2和算法节点3均依赖算法

节点1，算法节点4依赖算法节点2，算法节点5依赖算法节点3，算法节点6依赖算法节点4和算法节点5。

其中，目标任务可以是需要调用多种算法完成的数据处理任务，例如可以是图像处理任务或者音频处理任务等。

示例性的，获取目标任务对应的算法有向图的方式可以是：获取目标任务所需的多个处理算法；确定多个处理算法的依赖关系；基于依赖关系建立算法有向图。

其中，获取目标任务所需的多个处理算法的过程可以是：首先确定目标任务所需处理的对象（如图像或者音频）的初始状态以及最终状态，然后基于初始状态和最终状态将目标任务划分为多个阶段，然后确定出每个阶段所需调用的处理算法。确定多个处理算法的依赖关系的方式可以是：根据多个处理算法的执行顺序确定出处理算法间的依赖关系。基于依赖关系建立算法有向图的方式可以是：在具有相邻依赖关系的两个处理算法对应的算法节点添加有向边，有向边的起始端设置被依赖的算法节点，有向边的结束端设置依赖的算法节点。示例性的，一目标任务是对音频进行处理，其经历的阶段为：首先对音频进行滤波处理，然后对滤波后的音频进行音量检测，再然后对音频进行音量放大，最后对音频进行变调处理。从上述可以看出，该目标任务需要调用的算法包括：音频滤波算法、音量检测算法、音量放大算法及音频变调算法。且各算法的依赖关系为：音量检测算法依赖音频滤波算法的处理结果、音量放大算法依赖音量检测算法的处理结果，音频变调算法依赖音量放大算法的处理结果。因此确定出的算法有向图参见图3。本实施例的技术方案，基于处理算法间的依赖关系建立算法有向图，可以提高算法有向图的准确性。

S120，对算法有向图中的多个算法节点进行分组，获得多个节点组。

本实施例中，并行异步框架执行的基础首先在于找到算法有向图中可以并行执行的算法节点。从结构上来看，当算法有向图的深度与算法节点总数量相等时，此时没有任何两个算法节点存在并行执行的可能性。其中，算法有向图的深度可以是最长的链路中包含的算法节点的数量，当算法有向图的深度小于算法节点总数量时，执行本实施例的方案。

示例性的，需要在算法有向图中查找到可以并行处理的算法节点，将这些算法节点划分为一组，从而获得多个节点组。

可选的，对算法有向图中的多个算法节点进行分组的方式可以是：提取算法有向图中的数据交汇节点；将数据交汇节点的上游算法节点划分为一组，将数据交汇节点的下游算法节点划分为一组，将数据交汇节点作为一组，获得多

个节点组。

其中，数据交汇节点也可以称之为“瓶颈”节点，数据交汇节点的特点是其上游的算法节点均需通过该节点与其下游的算法节点进行联通。数据交汇节点的上游算法节点可以理解为处于该数据交汇节点与上一个数据交汇节点间的算法节点，数据交汇节点的下游算法节点可以理解为处于该数据交汇节点与下一个数据交汇节点间的算法节点。示例性的，当确定出多个数据交汇节点，将各数据交汇节点分别划分为一组，将相邻数据交汇节点之间的算法节点划分为一组。示例性的，图4是本实施例中对算法节点进行分组的示例图，如图4所示，节点1和节点4为数据交汇节点，节点2和节点3均位于节点1与节点4之间，节点5和节点5位于节点4的下游，则将节点1划分为1组，节点2和节点3划分为一组，节点4划分为1组，节点5和节点6划分为一组。需要注意的是，采用上述方案获得的节点组，节点组内仍然存在多种结构形式，因此还是需要任务调度器来对节点组内的节点进行筛选和调度，从而实现时序上的并行。本实施例的方案，基于数据交汇点对算法节点进行分组，可以提高分组的速度。

可选的，对算法有向图中的多个算法节点进行分组的方式可以是：获取算法有向图中各算法节点的深度；将深度相同的算法节点划分为一组，获得多个节点组。

其中，算法节点的深度可以理解为算法节点所处链路的深度，若算法节点处于多个链路上，则分别获取算法节点在各链路上的深度，将最大的深度确定为最终的深度。示例性的，获取各算法节点在算法有向图的最大深度，将最大深度相同的算法节点划分为一组，从而获得多个节点组。示例性的，图5是本实施例中对算法节点进行分组的示例图，如图5所示，节点2和节点3的深度相同，节点4、节点5和节点6的深度相同，节点7和节点8的深度相同，因此，需要将节点1划分为一组，节点2和节点3划分为一组，节点4、节点5和节点6划分为一组，节点7和节点8划分为一组。本实施例中，基于深度对算法节点进行分组，可以提高分组的准确性。

S130，串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法。

其中，串行调度多个节点组可以理解为依次调度各节点组。示例性的，串行调度多个节点组的方式可以是：若当前调度的节点组内的各算法节点对应的处理算法均执行完成，则继续调度下一个节点组。本实施例中，当前节点组的各算法节点对应的处理算法均执行完成后，才继续调度下一个节点组，使得目标任务的有序执行，从而保证结果的正确性。

可选的，串行调度多个节点组，并行调度节点组内的至少一个算法节点对

应的处理算法的方式可以是：对多个算法节点分别创建线程，并设置未被调度到的算法节点对应的线程处于等待状态；当调度到当前节点组时，启动当前节点组对应的线程，使得启动的线程执行当前节点组内的至少一个算法节点对应的处理算法。

本实施例中，对多个算法节点分别创建线程之后，整个算法有向图通过操作系统提供的等待（Wait）与启动（Signal）机制自动执行。对每一个算法节点，对前置节点的输入依赖使其在初始化之后自动进入等待状态，而执行完成的算法节点输出处理结果后会触发其后置算法节点线程的启动，在这种规则框架下，由于算法有向图的源节点（Source）不依赖于任何前置节点，将成为第一个被执行的节点，执行结束后将顺次触发多个后置节点的执行直至整个算法有向图执行结束。

示例性的，对多个算法节点分别创建线程之后，首先启动起始节点组对应的线程，并行执行起始节点组内至少一个算法节点对应的处理算法，后置节点组对应的线程进入等待状态。当起始节点组内的一个处理算法执行完成后，终止该处理算法的线程，当起始节点组内所有处理算法均执行完成后，启动下一个节点组对应的线程，使得启动线程并行执行当前节点组内的算法节点对应的处理算法，依次类推，直到整个算法有向图执行完成。本实施例中，对多个算法节点预先创建线程，然后整个过程由操作系统掌控完成，其优势在于执行过程可以针对不同平台上。

可选的，该方法还包括如下步骤：设置缓存池；将各算法节点对应的处理算法的处理结果存储至缓存池，使得后置算法节点从缓存池中读取处理结果进行处理。

本实施例中，组间信息可以通过统一的缓存池（BufferContainer）实现输入输出信息的共享，可以避免浪费计算资源。

可选的，串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法的方式可以是：当调度到当前节点组时，创建并启动当前节点组对应的线程，使得启动的线程执行当前节点组内的至少一个算法节点对应的处理算法；当算法节点对应的处理算法执行完成时，终止算法节点对应的线程，并将处理结果存入第一队列。

其中，第一队列可以用于存储算法节点的处理结果。本实施例中，通过任务调度器来实现线程的实施分配和中止，由线程池分配线程执行算法节点对应的处理算法。

其中，创建并启动当前节点组对应的线程，使得启动的线程执行当前节点

组内的至少一个算法节点对应的处理算法的过程可以是：从第一队列中读取处理结果，并根据处理结果确定当前节点组；将当前节点组包含的算法节点信息写入第二队列；基于第二队列中的算法节点信息创建并启动线程，使得启动的线程执行算法节点信息对应的处理算法。

其中，第二队列用于存储算法节点信息，即节点任务。

示例性的，图6是本实施例中算法调度示例图。如图6所示，任务调度器从第一队列中读取处理结果，根据算法有向图中的优先级，确定读取到的处理结果的后置节点组，将后置节点组的算法节点信息写入第二队列，线程池根据第一队列中的算法节点信息创建并分配线程来执行后置节点组内的处理算法，并将各处理算法的处理结果写入第一队列，直到算法有向图执行完成，将最终的结果输出。本实施例的方案，实施的创建启动线程以执行算法节点对应的处理算法，可以避免资源的浪费，并可以提高算法的执行效率。

本公开实施例的技术方案，获取目标任务对应的算法有向图；其中，算法有向图包含多个算法节点；对算法有向图中的多个算法节点进行分组，获得多个节点组；串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法。本公开实施例提供的调度方法，对算法有向图中的多个算法节点分组后，串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法，可以提高目标任务处理时的吞吐率及计算效率。

图7是本公开实施例提供的一种调度装置的结构示意图。如图7所示，该装置包括：

算法有向图获取模块210，设置为获取目标任务对应的算法有向图；其中，算法有向图包含多个算法节点；

算法节点分组模块220，设置为对算法有向图中的多个算法节点进行分组，获得多个节点组；

调度模块230，设置为串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法

可选的，算法有向图获取模块210，设置为通过以下方式建立算法有向图：

获取目标任务所需的多个处理算法；

确定多个处理算法的依赖关系；

基于依赖关系建立算法有向图。

可选的，算法节点分组模块220，设置为通过以下方式对算法有向图中的多个算法节点进行分组，获得多个节点组：

提取算法有向图中的数据交汇节点；

将数据交汇节点的上游算法节点划分为一组，将数据交汇节点的下游算法节点划分为一组，将数据交汇节点作为一组，获得多个节点组。

可选的，算法节点分组模块220，还设置为通过以下方式对算法有向图中的多个算法节点进行分组，获得多个节点组：

获取算法有向图中各算法节点的深度；

将深度相同的算法节点划分为一组，获得多个节点组。

可选的，调度模块230，设置为通过以下方式串行调度多个节点组：

若当前调度的节点组内的各算法节点对应的处理算法均执行完成，则继续调度下一个节点组。

可选的，调度模块230，设置为通过以下方式串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法：

对多个算法节点分别创建线程，并设置未被调度到的算法节点对应的线程处于等待状态；

当调度到当前节点组时，启动当前节点组对应的线程，使得启动的线程执行当前节点组内的至少一个算法节点对应的处理算法。

可选的，该装置还包括：缓存池设置模块，设置为：

设置缓存池；

将各算法节点对应的处理算法的处理结果存储至缓存池，使得后置算法节点从缓存池中读取处理结果进行处理。

可选的，调度模块230，设置为通过以下方式串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法：

当调度到当前节点组时，创建并启动当前节点组对应的线程，使得启动的线程执行当前节点组内的至少一个算法节点对应的处理算法；

当算法节点对应的处理算法执行完成时，终止算法节点对应的线程，并将处理结果存入第一队列。

可选的，调度模块230，设置为通过以下方式创建并启动当前节点组对应的线程，使得启动的线程执行当前节点组内的至少一个算法节点对应的处理算法：

从第一队列中读取处理结果，并根据处理结果确定当前节点组；

将当前节点组包含的算法节点信息写入第二队列；

基于第二队列中的算法节点信息创建并启动线程，使得启动的线程执行算法节点信息对应的处理算法。

上述装置可执行本公开前述所有实施例所提供的方法，具备执行上述方法相应的功能模块和有益效果。未在本实施例中详尽描述的技术细节，可参见本公开前述所有实施例所提供的方法。

下面参考图8，其示出了适于用来实现本公开实施例的电子设备300的结构示意图。本公开实施例中的电子设备可以包括诸如移动电话、笔记本电脑、数字广播接收器、个人数字助理（Personal Digital Assistant，PDA）、平板电脑（Portable Android Device，PAD）、便携式多媒体播放器（Portable Media Player，PMP）、车载终端（例如车载导航终端）等的移动终端以及诸如数字TV、台式计算机等的固定终端，或者各种形式的服务器，如独立服务器或者服务器集群。图8示出的电子设备仅仅是一个示例。

如图8所示，电子设备300可以包括处理装置（例如中央处理器、图形处理器等）301，处理装置301可以根据存储在只读存储装置（Read Only Memory，ROM）302中的程序或者从存储装置308加载到随机访问存储装置（Random Access Memory，RAM）303中的程序而执行各种适当的动作和处理。在RAM 303中，还存储有电子设备300操作所需的各种程序和数据。处理装置301、ROM 302以及RAM 303通过总线304彼此相连。输入/输出（Input/Output，I/O）接口305也连接至总线304。

通常，以下装置可以连接至I/O接口305：包括例如触摸屏、触摸板、键盘、鼠标、摄像头、麦克风、加速度计、陀螺仪等的输入装置306；包括例如液晶显示器（Liquid Crystal Display，LCD）、扬声器、振动器等的输出装置307；包括例如磁带、硬盘等的存储装置308；以及通信装置309。通信装置309可以允许电子设备300与其他设备进行无线或有线通信以交换数据。虽然图8示出了具有各种装置的电子设备300，但是应理解的是，并不要求实施或具备所有示出的装置。可以替代地实施或具备更多或更少的装置。

在一实施例中，根据本公开的实施例，上文参考流程图描述的过程可以被实现为计算机软件程序。例如，本公开的实施例包括一种计算机程序产品，其包括承载在计算机可读介质上的计算机程序，该计算机程序包含用于执行调度方法的程序代码。在这样的实施例中，该计算机程序可以通过通信装置309从网络上被下载和安装，或者从存储装置308被安装，或者从ROM 302被安装。在该计算机程序被处理装置301执行时，执行本公开实施例的方法中限定的上述功能。

需要说明的是，本公开上述的计算机可读介质可以是计算机可读信号介质或者计算机可读存储介质或者是上述两者的组合。计算机可读存储介质例如可

以是电、磁、光、电磁、红外线、或半导体的系统、装置或器件，或者以上的组合。计算机可读存储介质的示例可以包括：具有一个或多个导线的电连接、便携式计算机磁盘、硬盘、随机访问存储器（RAM）、只读存储器（ROM）、可擦式可编程只读存储器（如电子可编程只读存储器（Electronic Programable Read Only Memory，EPROM）或闪存）、光纤、便携式紧凑磁盘只读存储器（Compact Disc-Read Only Memory，CD-ROM）、光存储器件、磁存储器件、或者上述的合适的组合。在本公开中，计算机可读存储介质可以是包含或存储程序的有形介质，该程序可以被指令执行系统、装置或者器件使用或者与其结合使用。而在本公开中，计算机可读信号介质可以包括在基带中或者作为载波一部分传播的数据信号，其中承载了计算机可读的程序代码。这种传播的数据信号可以采用多种形式，包括电磁信号、光信号或上述的合适的组合。计算机可读信号介质还可以是计算机可读存储介质以外的任何计算机可读介质，该计算机可读信号介质可以发送、传播或者传输用于由指令执行系统、装置或者器件使用或者与其结合使用的程序。计算机可读介质上包含的程序代码可以用任何适当的介质传输，包括：电线、光缆、射频（Radio Frequency，RF）等，或者上述的合适的组合。

在一些实施方式中，客户端、服务器可以利用诸如超文本传输协议（HyperText Transfer Protocol，HTTP）之类的任何当前已知或未来研发的网络协议进行通信，并且可以与任意形式或介质的数字数据通信（例如，通信网络）互连。通信网络的示例包括局域网（Local Area Network，LAN），广域网（Wide Area Network，WAN），网际网（例如，互联网）以及端对端网络（例如，ad hoc 端对端网络），以及当前已知或未来研发的网络。

上述计算机可读介质可以是上述电子设备中所包含的；也可以是单独存在，而未装配入该电子设备中。

上述计算机可读介质承载有至少一个程序，当上述至少一个程序被该电子设备执行时，使得该电子设备：获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；对所述算法有向图中的多个算法节点进行分组，获得多个节点组；串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

可以以一种或多种程序设计语言或其组合来编写用于执行本公开的操作的计算机程序代码，上述程序设计语言包括面向对象的程序设计语言诸如Java、Smalltalk、C++，还包括常规的过程式程序设计语言诸如“C”语言或类似的程序设计语言。程序代码可以完全地在用户计算机上执行、部分地在用户计算机上执行、作为一个独立的软件包执行、部分在用户计算机上部分在远程计算机上

执行、或者完全在远程计算机或服务器上执行。在涉及远程计算机的情形中，远程计算机可以通过任意种类的网络包括局域网(LAN)或广域网(WAN)连接到用户计算机，或者，可以连接到外部计算机（例如利用因特网服务提供商来通过因特网连接）。

附图中的流程图和框图，图示了按照本公开各种实施例的系统、方法和计算机程序产品的可能实现的体系架构、功能和操作。在这点上，流程图或框图中的每个方框可以代表一个模块、程序段、或代码的一部分，该模块、程序段、或代码的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。也应当注意，在有些作为替换的实现中，方框中所标注的功能也可以以不同于附图中所标注的顺序发生。例如，两个接连地表示的方框实际上可以基本并行地执行，它们有时也可以按相反的顺序执行，这依所涉及的功能而定。也要注意的是，框图和/或流程图中的每个方框、以及框图和/或流程图中的方框的组合，可以用执行规定的功能或操作的专用的基于硬件的系统来实现，或者可以用专用硬件与计算机指令的组合来实现。

描述于本公开实施例中所涉及到的单元可以通过软件的方式实现，也可以通过硬件的方式来实现。其中，单元的名称在某种情况下并不构成对该单元本身的限定。

本文中以上描述的功能可以至少部分地由至少一个硬件逻辑部件来执行。例如，可以使用的示范类型的硬件逻辑部件包括：现场可编程门阵列（Field-Programmable Gate Array，FPGA）、专用集成电路（Application Specific Integrated Circuit，ASIC）、专用标准产品（Application Specific Standard Parts，ASSP）、片上系统（System on Chip，SOC）、复杂可编程逻辑设备（Complex Programmable Logic Device，CPLD）等。

在本公开的上下文中，机器可读介质可以是有形的介质，其可以包含或存储以供指令执行系统、装置或设备使用或与指令执行系统、装置或设备结合地使用的程序。机器可读介质可以是机器可读信号介质或机器可读储存介质。机器可读介质可以包括电子的、磁性的、光学的、电磁的、红外的、或半导体系统、装置或设备，或者上述内容的合适组合。机器可读存储介质的示例可包括基于一个或多个线的电气连接、便携式计算机盘、硬盘、随机存取存储器(RAM)、只读存储器（ROM）、可擦除可编程只读存储器（EPROM或快闪存储器）、光纤、便捷式紧凑盘只读存储器（CD-ROM）、光学储存设备、磁储存设备、或上述内容的合适组合。

根据本公开实施例的一个或多个实施例，本公开实施例公开了一种调度方法，包括：

获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；

对所述算法有向图中的多个算法节点进行分组，获得多个节点组；

串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

可选地，获取目标任务对应的算法有向图，包括：

获取所述目标任务所需的多个处理算法；

确定所述多个处理算法的依赖关系；

基于所述依赖关系建立算法有向图。

可选地，对所述算法有向图中的多个算法节点进行分组，包括：

提取所述算法有向图中的数据交汇节点；

将所述数据交汇节点的上游算法节点划分为一组，将所述数据交汇节点的下游算法节点划分为一组，将所述数据交汇节点作为一组，获得多个节点组。

可选地，对所述算法有向图中的多个算法节点进行分组，包括：

获取所述算法有向图中各算法节点的深度；

将深度相同的算法节点划分为一组，获得多个节点组。

可选地，串行调度所述多个节点组，包括：

若当前调度的节点组内的各算法节点对应的处理算法均执行完成，则继续调度下一个节点组。

可选地，串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法，包括：

对所述多个算法节点分别创建线程，并设置未被调度到的算法节点对应的线程处于等待状态；

当调度到当前节点组时，启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法。

可选地，所述方法还包括：

设置缓存池；

将各算法节点对应的处理算法的处理结果存储至所述缓存池，使得后置算法节点从所述缓存池中读取所述处理结果进行处理。

可选地，串行调度所述多个节点组，并行调度所述节点组内的至少一个算

法节点对应的处理算法，包括：

当调度到当前节点组时，创建并启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法；

当算法节点对应的处理算法执行完成时，终止所述算法节点对应的线程，并将处理结果存入第一队列。

可选地，创建并启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法，包括：

从所述第一队列中读取处理结果，并根据所述处理结果确定当前节点组；

将所述当前节点组包含的算法节点信息写入第二队列；

基于所述第二队列中的算法节点信息创建并启动线程，使得启动的线程执行所述算法节点信息对应的处理算法。

应该理解，可以使用上面所示的各种形式的流程，重新排序、增加或删除步骤。例如，本公开中记载的各步骤可以并行地执行也可以顺序地执行也可以不同的次序执行，只要能够实现本公开的技术方案所期望的结果。

权 利 要 求 书

1. 一种调度方法，包括：

获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；

对所述算法有向图中的多个算法节点进行分组，获得多个节点组；

串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

2. 根据权利要求 1 所述的方法，其中，所述获取目标任务对应的算法有向图，包括：

获取所述目标任务所需的多个处理算法；

确定所述多个处理算法的依赖关系；

基于所述依赖关系建立算法有向图。

3. 根据权利要求 1 所述的方法，其中，所述对所述算法有向图中的多个算法节点进行分组，包括：

提取所述算法有向图中的数据交汇节点；

将所述数据交汇节点的上游算法节点划分为一组，将所述数据交汇节点的下游算法节点划分为一组，将所述数据交汇节点作为一组，获得多个节点组。

4. 根据权利要求 1 所述的方法，其中，所述对所述算法有向图中的多个算法节点进行分组，包括：

获取所述算法有向图中的多个算法节点的深度；

将深度相同的算法节点划分为一组，获得多个节点组。

5. 根据权利要求 1 所述的方法，其中，所述串行调度所述多个节点组，包括：

响应于当前调度的节点组内的所述至少一个算法节点对应的处理算法均执行完成，继续调度下一个节点组。

6. 根据权利要求 1 或 5 所述的方法，其中，所述串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法，包括：

对所述多个算法节点分别创建线程，并设置未被调度到的算法节点对应的线程处于等待状态；

当调度到当前节点组时，启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法。

7. 根据权利要求 6 所述的方法，所述方法还包括：

设置缓存池；

将所述至少一个算法节点对应的处理算法的处理结果存储至所述缓存池，使得后置算法节点从所述缓存池中读取所述处理结果进行处理。

8. 根据权利要求 1 所述的方法，其中，所述串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法，包括：

当调度到当前节点组时，创建并启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法；

当算法节点对应的处理算法执行完成时，终止所述算法节点对应的线程，并将处理结果存入第一队列。

9. 根据权利要求 8 所述的方法，其中，所述创建并启动所述当前节点组对应的线程，使得启动的线程执行所述当前节点组内的至少一个算法节点对应的处理算法，包括：

从所述第一队列中读取处理结果，并根据所述处理结果确定当前节点组；

将所述当前节点组包含的算法节点信息写入第二队列；

基于所述第二队列中的算法节点信息创建并启动线程，使得启动的线程执行所述算法节点信息对应的处理算法。

10. 一种调度装置，包括：

算法有向图获取模块，设置为获取目标任务对应的算法有向图；其中，所述算法有向图包含多个算法节点；

算法节点分组模块，设置为对所述算法有向图中的多个算法节点进行分组，获得多个节点组；

调度模块，设置为串行调度所述多个节点组，并行调度所述节点组内的至少一个算法节点对应的处理算法。

11. 一种电子设备，所述电子设备包括：

一个或多个处理装置；

存储装置，设置为存储一个或多个程序；

当所述一个或多个程序被所述一个或多个处理装置执行，使得所述一个或多个处理装置实现如权利要求 1-9 中任一所述的调度方法。

12. 一种计算机可读介质，所述计算机可读介质上存储有计算机程序，所述

计算机程序被处理装置执行时实现如权利要求1-9中任一所述的调度方法。

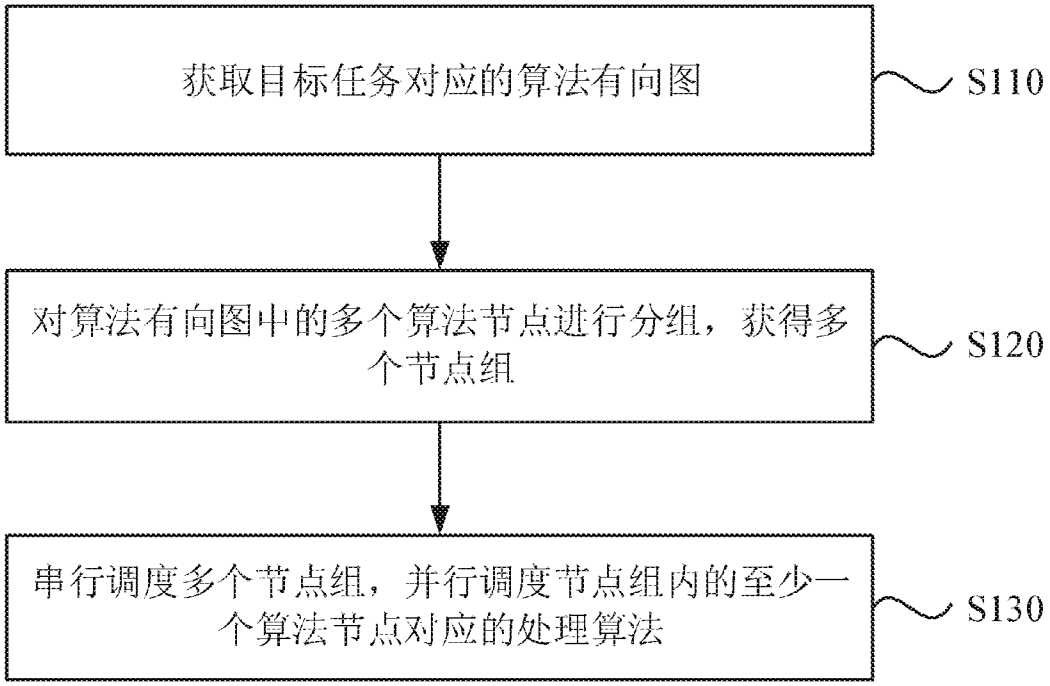
获取目标任务对应的算法有向图
S110
对算法有向图中的多个算法节点进行分组，获得多个节点组
S120
串行调度多个节点组，并行调度节点组内的至少一个算法节点对应的处理算法
S130

图 1

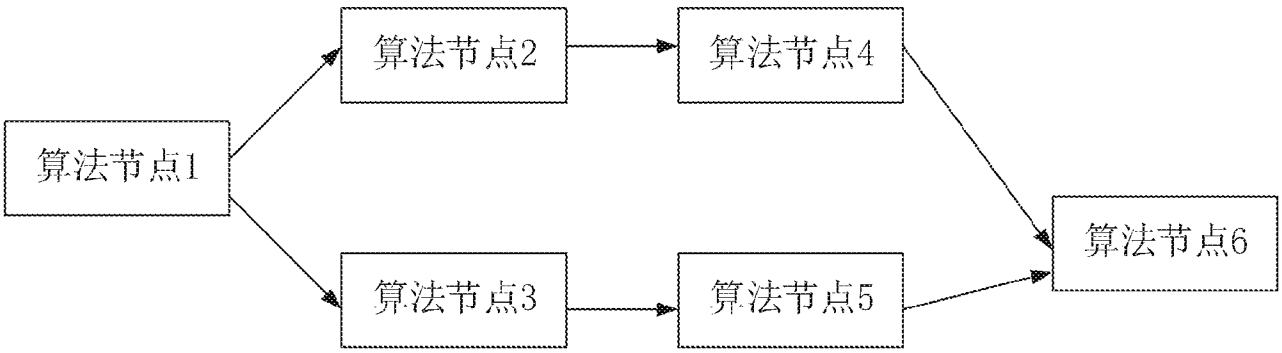
算法节点1
算法节点2
算法节点3
算法节点4
算法节点5
算法节点6

图 2

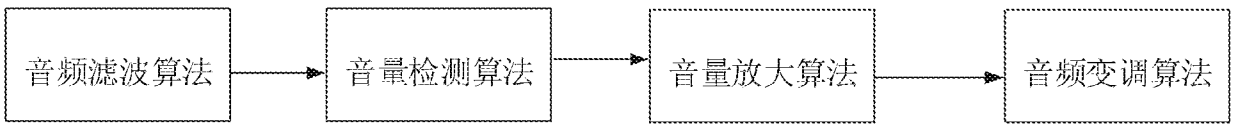
音频滤波算法
音量检测算法
音量放大算法
音频变调算法

图 3

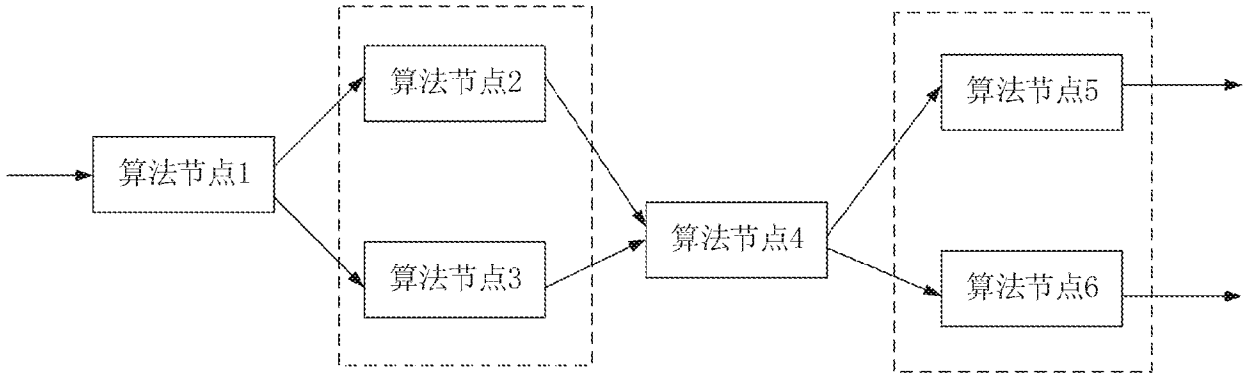
算法节点1
算法节点2
算法节点3
算法节点4
算法节点5
算法节点6

图 4

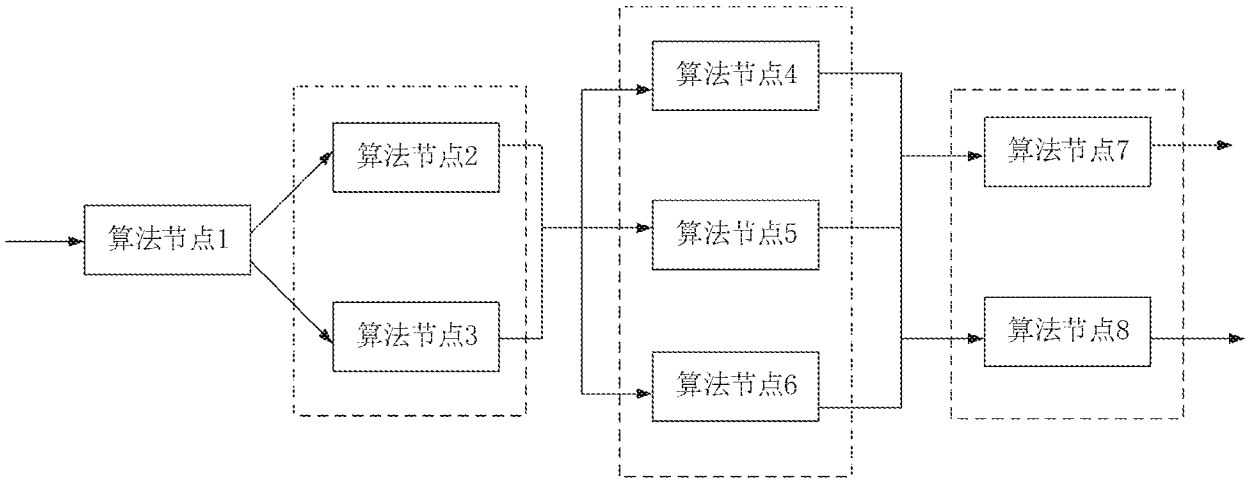
算法节点1
算法节点2
算法节点3
算法节点4
算法节点5
算法节点6
算法节点7
算法节点8

图 5

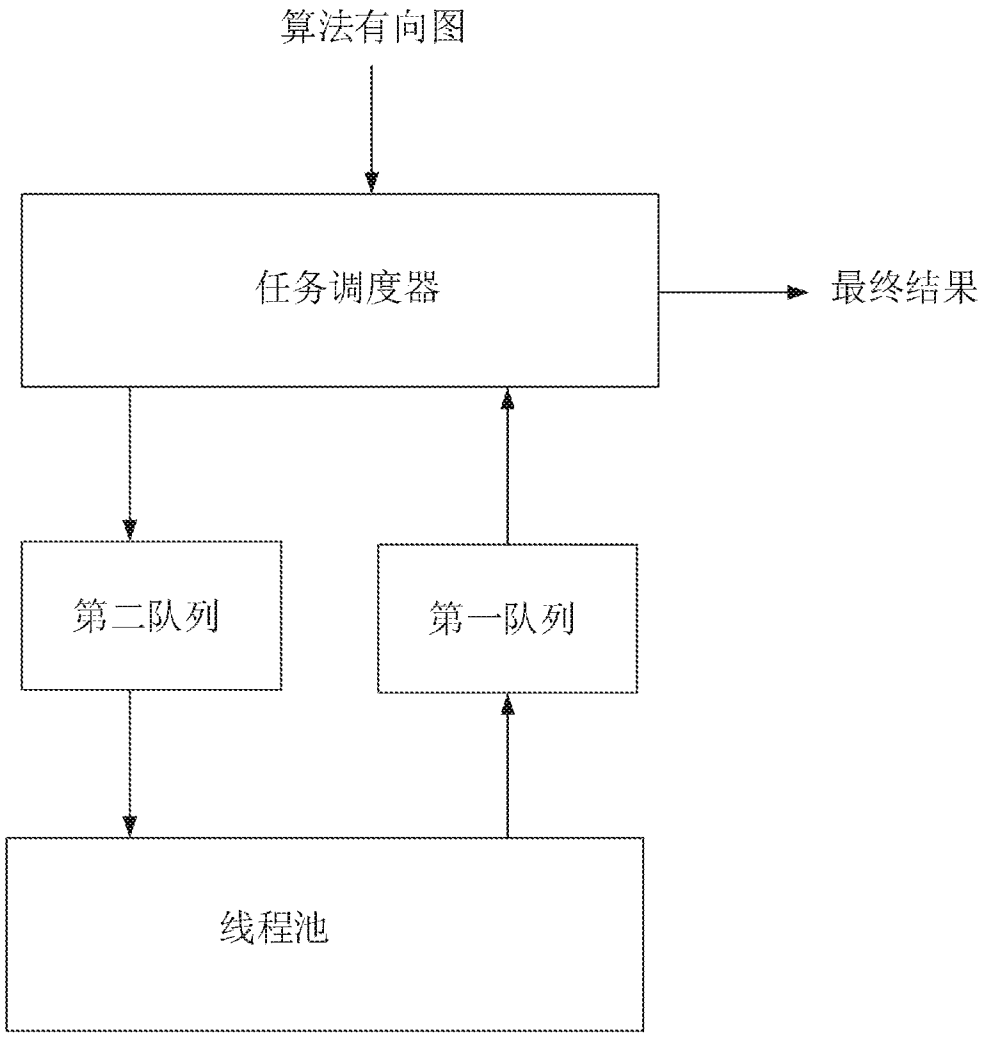
算法有向图
任务调度器
最终结果
第二队列
第一队列
线程池

图 6

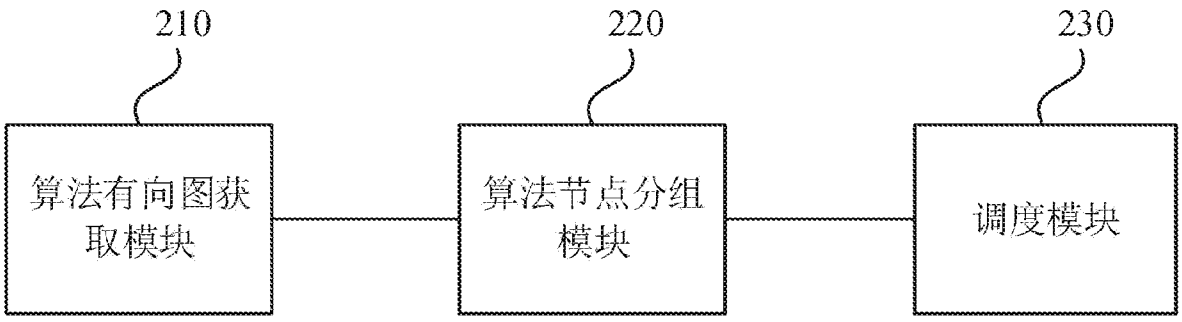
210
220
230
算法有向图获取模块
算法节点分组模块
调度模块

图 7

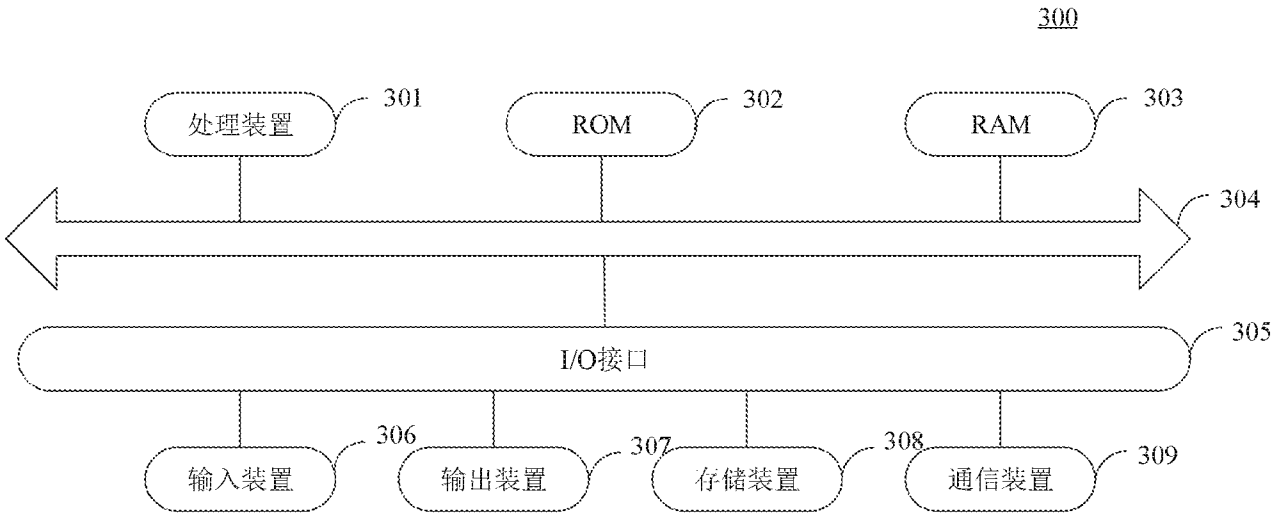
300
处理装置
301
ROM
302
RAM
303
304
I/O接口
305
输入装置
306
输出装置
307
存储装置
308
通信装置
309

图 8