说 明 书 摘 要

本发明公开了一种高能效软硬协同图计算方法及装置，该方法主要包括一种新颖的可编程硬件加速器设计和一种高效的图计算软件设计。该可编程硬件加速器设计，主要包括一个依赖路径预取单元和一个直接依赖管理单元。该加速器可集成到多核处理器中，通过有效实施本发明提出的图计算软件设计方法，实现高效图处理。该图计算软件设计，主要包括图数据预处理阶段和依赖驱动的异步图处理阶段。在图数据预处理阶段，首先提取出图数据的中心顶点和中心路径，然后在此基础上划分核心子图、核心路径和核心顶点；在依赖驱动的异步图处理阶段，通过依赖路径预取单元从活跃顶点开始沿依赖路径预取边进行处理，同时通过直接依赖管理单元将核心依赖路径首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理，在路径首顶点状态更新后，根据直接依赖公式快速计算出对路径尾顶点的影响并更新尾顶点状态，而不需要等待路径首顶点状态经过多轮迭代传播到尾顶点，从而可以在多个核上高并发地对多条路径进行处理，加速图顶点状态的传播，提高图计算的收敛速度。

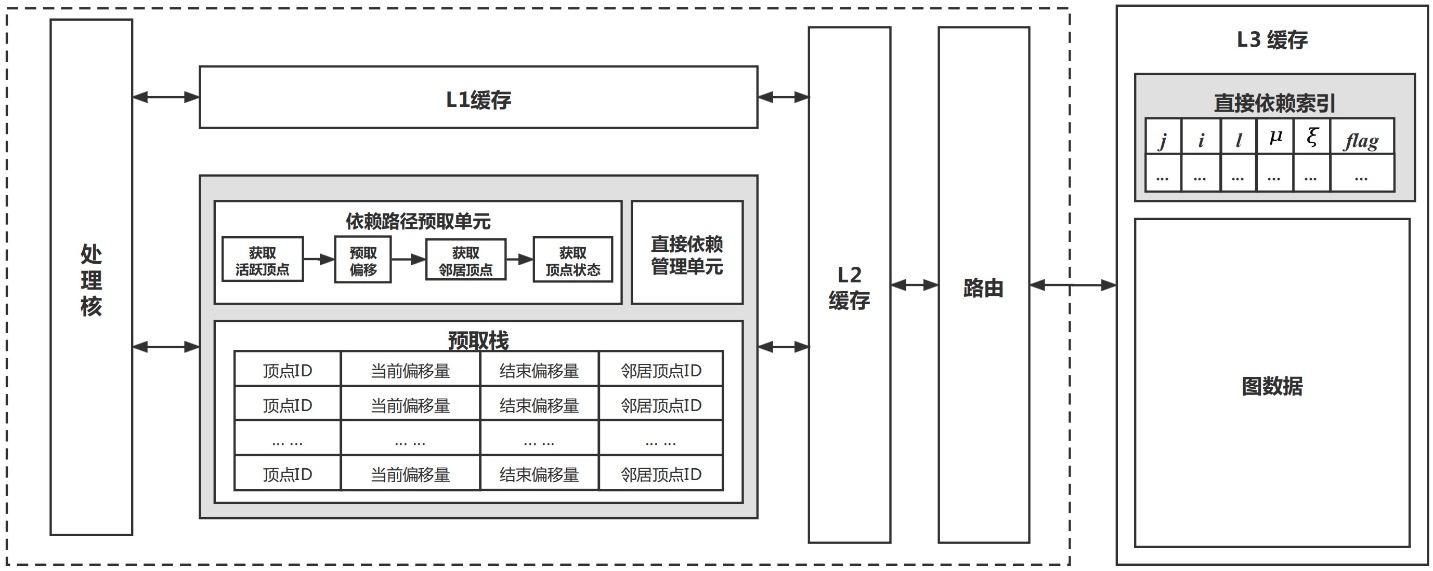
摘 要 附 图

图1 硬件加速器架构

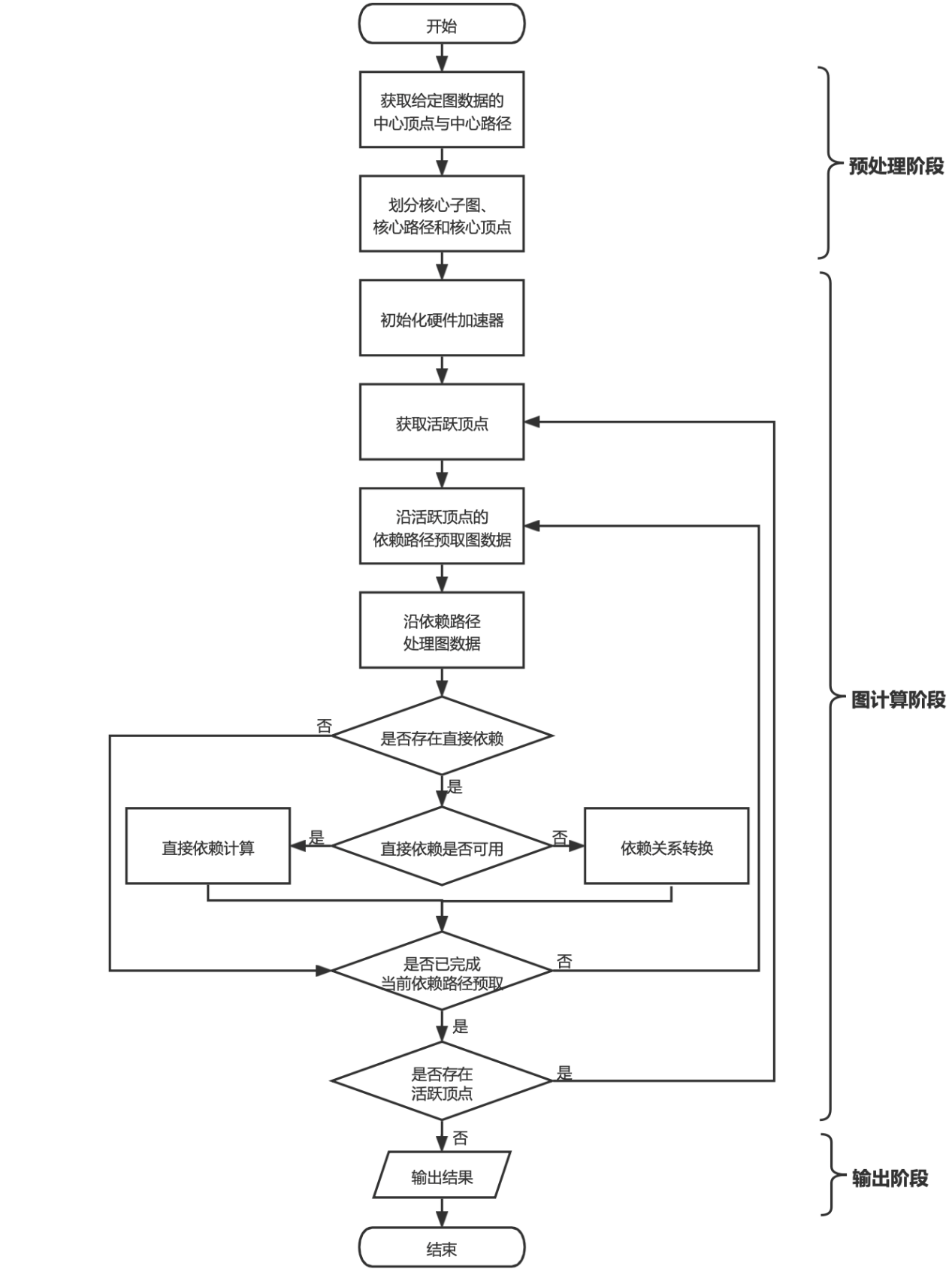


图2 图计算软件设计方法流程图

权 利 要 求 书

1、一种高能效软硬协同图计算方法及装置中的可编程硬件加速器设计，包括以下两个部分：

（1）依赖路径预取单元：用于从活跃顶点开始沿依赖路径预取待处理的边。

（2）直接依赖管理单元：用于将核心依赖路径上首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理。

2、一种高能效软硬协同图计算方法及装置中的图计算软件设计，包括以下步骤：

（1）图数据预处理。获取给定图的中心顶点与中心路径，并在此基础上划分核心子图、核心路径和核心顶点；

（2）依赖驱动的异步图处理。通过硬件加速器的依赖路径预取单元从活跃顶点开始沿依赖路径预取边进行处理，同时通过硬件加速器的直接依赖管理单元将核心依赖路径首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理，在路径首顶点状态更新后，根据直接依赖公式快速计算出对路径尾顶点的影响并更新尾顶点状态。

3、如权利要求2所述的方法，其特征在于，所述步骤（1）包括如

下子步骤：

（1.1）遍历图顶点，取大于度数阈值的顶点为中心顶点。度数阈值的计算方法为：根据用户指定的中心顶点的比例计算出中心顶点数量（为所有顶点数量），然后根据顶点度数对所有顶点进行降序排序，取第个顶点的度数为度数阈值。因对所有顶点进行排序的代价太高，所以也可以使用采样方法快速确定度数阈值，取占比为的采样顶点并对其降序排序，取第个顶点的度数为度数阈值。

（1.2）遍历图数据，基于中心顶点获取中心路径，即首尾顶点均为中心顶点的路径，中心路径的并集构成核心子图。具体而言，从步骤（1.1）中划分出的中心顶点集获取中心顶点，以此为根顶点执行深度优先搜索遍历，遍历过程中优先访问度数高的顶点，并指定遍历深度（默认深度为16），设遍历路径的首顶点为，尾顶点为，若属于中心顶点集，将顶点标记为已访问并将路径加入集合，若中心顶点集中的顶点均被标记为已访问则直接结束遍历。当前遍历完成后，若中心顶点集中的顶点未被全部访问，则选取中心顶点集中的下一个未访问顶点作为根顶点继续执行上述遍历，直至中心顶点集中的顶点均被访问，最终获得所有中心路径即为核心子图。

（1.3）遍历核心子图，将核心子图划分为相交顶点为路径首尾顶点的路径，称为核心路径。同时获取两条核心路径相交的顶点，称为核心顶点。具体而言，从步骤（1.2）中划分出的核心子图获取具有未访问边的顶点（优先选取中心顶点），从该顶点出发沿深度优先搜索遍历的顺序获取一条路径（路径最大长度默认为16），并将路径的首尾顶点加入核心顶点集合，重复上述步骤，直至核心子图的所有边都被访问。

4、如权利要求2所述的方法，其特征在于，所述步骤（2）包括如

下子步骤：

（2.1）硬件加速器初始化。通过调用加速器配置接口将图数据信息传递到加速器可访问的内存映射寄存器来初始化该加速器。默认传递的图数据信息包括：（a）CSR格式图数据所包括的偏移数组、边数组和顶点状态数组的大小和首地址；（b）分配给对应处理核的图分区的起始顶点与结束顶点；（c）图分区中核心顶点集合的大小和首地址；（d）对应处理核的本地循环队列的大小和首地址，该循环队列用于存储图分区中待处理的活跃顶点。

（2.2）依赖驱动的边预取。在每一轮图处理中，硬件加速器的依赖路径预取单元从本地循环队列中的活动顶点开始，通过深度优先搜索为其对应的处理核动态预取图分区的边。依赖路径预取单元使用一个固定深度的栈记录预取信息，栈中的每个条目都包含以下信息：（a）遍历中所访问顶点的；（b）该顶点的未访问边的当前偏移量与结束偏移量；（c）该顶点的未访问邻居顶点的。

硬件加速器的依赖路径预取单元以四段流水线形式完成边的预取，每次都将获取的边以及该边对应的一对顶点的状态输出到边缓冲区供处理核访问处理：

（2.2.1）若栈为空，则从本地循环队列中获取一个活动顶点，并将其压栈。

（2.2.2）从偏移数组获取栈顶顶点的出边起始/结束偏移。

（2.2.3）根据该顶点的未访问边获取未访问邻居顶点的，并将其中一个邻居顶点压栈。

（2.2.4）从顶点状态数组获取相关顶点的状态，将该边以及该边对应的一对顶点的状态输出到边缓冲区。若栈顶顶点属于顶点集合，则栈顶顶点出栈，并将其插入本地循环队列作为新的活跃顶点，进入步骤（2.2.1）。若无法从栈顶顶点的邻居中获取到图分区中的未访问顶点，则栈顶顶点出栈，进入步骤（2.2.1）。

（2.3）图顶点依赖关系转换。在依赖路径预取单元进行边预取的同时，直接依赖管理单元将核心依赖路径首尾顶点的间接依赖转换为直接依赖。当顶点之间的依赖关系为线性关系时，两个顶点的直接依赖关系可以表示为公式，其中、为顶点、的状态值，、为常数参数。直接依赖管理单元使用直接依赖索引数组保存路径首尾顶点间的直接依赖索引，数组中的每一项索引都包括首顶点编号、尾顶点编号、路径标识、参数、参数和索引标识。其中索引标识代表该索引的当前状态，一共分为三种情况：（a）索引标识为，则该索引无效；（b）索引标识为，则参数、参数的当前值为顶点、的一组状态值、；（c）索引标识为，则索引有效，参数、参数的值为直接依赖公式的参数值。依赖转换过程如下：

（2.3.1）直接依赖索引的索引标识初始化为，图处理过程中第一次完成对核心依赖路径（首尾顶点都属于顶点集的路径）的处理时，将其首尾顶点的编号、以及状态值、保存至直接依赖索引数组，并将索引标识设置为。

（2.3.2）第二次完成对路径的处理时，获得首尾顶点一组新的状态值，结合已存储在该索引、处的上一组状态值，连立代入直接依赖公式计算出常数参数、的值，将和的值保存至索引、处，并将索引标识设置为。

在后续的处理中，对路径首顶点进行更新后可以通过直接依赖索引获取直接依赖公式的参数，然后根据公式计算出首顶点更新对尾顶点的影响并更新尾顶点状态。

5、如权利要求4所述的间接依赖转换为直接依赖的过程，其特征在于，所述的图算法需要满足以下两个属性：

（1）属性1：图算法可使用Gather-Apply-Scatter(GAS)模型表示。

（2）属性2：图算法的边处理函数是一个线性表达式，通常表示为乘法或加法。

6、如权力要求4所述的图顶点依赖关系转换所产生的直接依赖索引，其特征在于，直接依赖管理单元使用定制化缓存策略对其进行缓存管理:

（1）依赖索引的可重用性划分。首先根据依赖源顶点的度数降序排列所有依赖索引，排列开始处的一个LLC大小的区域被划分为高重用区域，在高重用区域后的一个LLC大小的区域被划分为中等重用区域，剩余区域则被划分为低重用区域，每个区域中的依赖索引拥有对应等级的可重用性。

（2）定制化缓存策略包括插入策略、命中提升策略以及驱逐策略，具体如下：

（2.1）插入策略：当被访问的依赖索引未被缓存时，则将索引插入缓存，并根据索引所处的可重用区域设置不同的缓存优先级，否则不插入该索引。高重用区域的索引被设置为高优先级，避免被马上替换出缓存；中等重用区域的索引被设置为中等优先级，使得中等访问频率的索引在不导致缓存抖动的情况下也能有机会被重用；而低重用区域的索引与图数据则被设置为低优先级，使得它们能够立即被替换出缓存，从而更有效地利用缓存空间。

（2.2）命中提升策略：当依赖索引被命中时，升级该索引的缓存优先级。高重用区域的索引被立即升级为最高优先级，因为它们接下来很可能会被并发任务连续访问，而位于中等重用区域和低重用区域的索引则只向上升一级，因为位于这两个区域的索引的可重用性有限，逐步提升使得它们能够有机会被重用，同时又能不长时间地占用缓存空间。

（2.3）驱逐策略：当缓存空间已满时，最低缓存优先级的依赖索引或图数据将被优先替换出缓存。同时长时间未被命中的依赖索引将逐步降级，使得缓存管理能够更加灵活有效。

说 明 书

一种高能效软硬协同图计算方法及装置

**技术领域**

本发明涉及大数据处理的图计算技术领域，具体涉及一种高能效软硬协同图计算方法及装置。

**背景技术**

随着大数据时代的来临，图作为一种能够良好表达数据关联性的数据结构，在互联网应用、数据挖掘、科学计算等许多领域都得到了广泛的应用。现有的许多重要图应用都使用迭代图算法对图数据进行迭代处理直至收敛，例如路径分析、商品推荐、社交网络分析等。

在迭代图算法中，图顶点的状态更新依赖于其相邻图顶点的状态值，这种图结构的依赖性使得图顶点之间往往存在较长的依赖链，每个顶点的新状态需要沿依赖路径传播多轮才能到达其间接邻居。当沿依赖路径在多个处理核间进行顶点状态传播时会产生高额的同步开销。许多顶点在其邻居的新状态到达之前处于非活跃状态。此外，图顶点的过时状态可能会被其邻居读取，从而导致不必要的顶点状态更新。因此，多核处理器往往只能以较低的并行度在图数据上执行迭代图算法的有效更新，这严重影响了图处理效率。

为给图应用提供实时的结果，现有研究提出了许多软硬件解决方案来加速多核处理器上的图处理速度，但由于忽略了顶点间的依赖关系，现有解决方案仍存在对多核处理器利用不充分的问题，无法在图拓扑结构中高效地传播顶点状态，从而导致迭代图算法收敛速度缓慢。

**发明内容**

针对现有解决方案的不足，本发明提出一种高能效软硬协同图计算方法及装置，其目的在于解决迭代图处理在多核处理器上有效并行度低、图顶点状态传播缓慢、图处理效率低的问题。

为实现上述目的，本发明包括一种可编程硬件加速器设计方法和一种图计算软件设计方法，其中可编程硬件加速器设计包括以下两个部分：

（1）依赖路径预取单元：用于从活跃顶点开始沿依赖路径预取待处理的边。

（2）直接依赖管理单元：用于将核心依赖路径上首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理。

图计算软件设计方法在硬件加速器的基础上实现，包括以下步骤：

（1）图数据预处理。获取给定图的中心顶点与中心路径，并在此基础上划分核心子图、核心路径和核心顶点；

（2）依赖驱动的异步图处理。通过硬件加速器的依赖路径预取单元从活跃顶点开始沿依赖路径预取边进行处理，同时通过硬件加速器的直接依赖管理单元将核心依赖路径首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理，在路径首顶点状态更新后，根据直接依赖公式快速计算出对路径尾顶点的影响并更新尾顶点状态。

本发明中，将度数大于阈值的顶点定义为中心顶点；在定义中心顶点的基础上，将首尾顶点都是中心顶点的路径划分为中心路径；中心路径的并集构成核心子图；遍历核心子图，将其划分为相交顶点为路径首尾顶点的路径，称之为核心路径；核心路径相交的顶点，称之为核心顶点。由于现实世界中图的幂律特性，少数图顶点连接图的大部分边，因此绝大多数图顶点之间的状态传播都需要通过核心依赖路径进行，本发明的关键设计就是将核心依赖路径首尾顶点的间接依赖转换为直接依赖，从而并行化依赖路径上的异步顶点状态传播，以加速迭代图算法的收敛。

图计算软件设计方法的步骤（1）图数据预处理包括如下子步骤：

（1.1）遍历图顶点，取大于度数阈值的顶点为中心顶点。度数阈值的计算方法为：根据用户指定的中心顶点的比例计算出中心顶点数量（为所有顶点数量），然后根据顶点度数对所有顶点进行降序排序，取第个顶点的度数为度数阈值。因对所有顶点进行排序的代价太高，所以也可以使用采样方法快速确定度数阈值，取占比为的采样顶点并对其降序排序，取第个顶点的度数为度数阈值。

（1.2）遍历图数据，基于中心顶点获取中心路径，即首尾顶点均为中心顶点的路径，中心路径的并集构成核心子图。具体而言，从步骤（1.1）中划分出的中心顶点集获取中心顶点，以此为根顶点执行深度优先搜索遍历，遍历过程中优先访问度数高的顶点，并指定遍历深度（默认深度为16），设遍历路径的首顶点为，尾顶点为，若属于中心顶点集，将顶点标记为已访问并将路径加入集合，若中心顶点集中的顶点均被标记为已访问则直接结束遍历。当前遍历完成后，若中心顶点集中的顶点未被全部访问，则选取中心顶点集中的下一个未访问顶点作为根顶点继续执行上述遍历，直至中心顶点集中的顶点均被访问，最终获得所有中心路径即为核心子图。

（1.3）遍历核心子图，将核心子图划分为相交顶点为路径首尾顶点的路径，称为核心路径。同时获取两条核心路径相交的顶点，称为核心顶点。具体而言，从步骤（1.2）中划分出的核心子图获取具有未访问边的顶点（优先选取中心顶点），从该顶点出发沿深度优先搜索遍历的顺序获取一条路径（路径最大长度默认为16），并将路径的首尾顶点加入核心顶点集合，重复上述步骤，直至核心子图的所有边都被访问。

图计算软件设计方法的步骤（2）依赖驱动的异步图处理通过本发明提出的硬件加速器来有效实现。该硬件加速器的设计架构如图3所示，主要包括依赖路径预取单元和直接依赖管理单元，每个加速器与多核处理器的一个核心耦合，并通过二级缓存访问内存。基于硬件加速器实现的依赖驱动异步图处理，其主要包括如下子步骤：

（2.1）硬件加速器初始化。通过调用加速器配置接口将图数据信息传递到加速器可访问的内存映射寄存器来初始化该加速器。默认传递的图数据信息包括：（a）CSR格式图数据所包括的偏移数组、边数组和顶点状态数组的大小和首地址；（b）分配给对应处理核的图分区的起始顶点与结束顶点；（c）图分区中核心顶点集合的大小和首地址；（d）对应处理核的本地循环队列的大小和首地址，该循环队列用于存储图分区中待处理的活跃顶点。

（2.2）依赖驱动的边预取。在每一轮图处理中，硬件加速器的依赖路径预取单元从本地循环队列中的活动顶点开始，通过深度优先搜索为其对应的处理核动态预取图分区的边。依赖路径预取单元使用一个固定深度的栈记录预取信息，栈中的每个条目都包含以下信息：（a）遍历中所访问顶点的；（b）该顶点的未访问边的当前偏移量与结束偏移量；（c）该顶点的未访问邻居顶点的。

硬件加速器的依赖路径预取单元以四段流水线形式完成边的预取，每次都将获取的边以及该边对应的一对顶点的状态输出到边缓冲区供处理核访问处理：

（2.2.1）若栈为空，则从本地循环队列中获取一个活动顶点，并将其压栈。

（2.2.2）从偏移数组获取栈顶顶点的出边起始/结束偏移。

（2.2.3）根据该顶点的未访问边获取未访问邻居顶点的，并将其中一个邻居顶点压栈。

（2.2.4）从顶点状态数组获取相关顶点的状态，将该边以及该边对应的一对顶点的状态输出到边缓冲区。若栈顶顶点属于顶点集合，则栈顶顶点出栈，并将其插入本地循环队列作为新的活跃顶点，进入步骤（2.2.1）。若无法从栈顶顶点的邻居中获取到图分区中的未访问顶点，则栈顶顶点出栈，进入步骤（2.2.1）。

（2.3）图顶点依赖关系转换。在依赖路径预取单元进行边预取的同时，直接依赖管理单元将核心依赖路径首尾顶点的间接依赖转换为直接依赖。当顶点之间的依赖关系为线性关系时，两个顶点的直接依赖关系可以表示为公式，其中、为顶点、的状态值，、为常数参数。直接依赖管理单元使用直接依赖索引数组保存路径首尾顶点间的直接依赖索引，数组中的每一项索引都包括首顶点编号、尾顶点编号、路径标识、参数、参数和索引标识。其中索引标识代表该索引的当前状态，一共分为三种情况：（a）索引标识为，则该索引无效；（b）索引标识为，则参数、参数的当前值为顶点、的一组状态值、；（c）索引标识为，则索引有效，参数、参数的值为直接依赖公式的参数值。依赖转换过程如下：

（2.3.1）直接依赖索引的索引标识初始化为，图处理过程中第一次完成对核心依赖路径（首尾顶点都属于顶点集的路径）的处理时，将其首尾顶点的编号、以及状态值、保存至直接依赖索引数组，并将索引标识设置为。

（2.3.2）第二次完成对路径的处理时，获得首尾顶点一组新的状态值，结合已存储在该索引、处的上一组状态值，连立代入直接依赖公式计算出常数参数、的值，将和的值保存至索引、处，并将索引标识设置为。

在后续的处理中，对路径首顶点进行更新后可以通过直接依赖索引获取直接依赖公式的参数，然后根据公式计算出首顶点更新对尾顶点的影响并更新尾顶点状态。此外，直接依赖管理单元使用定制化缓存策略对依赖索引进行缓存管理：

（1）依赖索引的可重用性划分。首先根据依赖源顶点的度数降序排列所有依赖索引，排列开始处的一个LLC大小的区域被划分为高重用区域，在高重用区域后的一个LLC大小的区域被划分为中等重用区域，剩余区域则被划分为低重用区域，每个区域中的依赖索引拥有对应等级的可重用性。

（2）定制化缓存策略包括插入策略、命中提升策略以及驱逐策略，具体如下：

（2.1）插入策略：当被访问的依赖索引未被缓存时，则将索引插入缓存，并根据索引所处的可重用区域设置不同的缓存优先级，否则不插入该索引。高重用区域的索引被设置为高优先级，避免被马上替换出缓存；中等重用区域的索引被设置为中等优先级，使得中等访问频率的索引在不导致缓存抖动的情况下也能有机会被重用；而低重用区域的索引与图数据则被设置为低优先级，使得它们能够立即被替换出缓存，从而更有效地利用缓存空间。

（2.2）命中提升策略：当依赖索引被命中时，升级该索引的缓存优先级。高重用区域的索引被立即升级为最高优先级，因为它们接下来很可能会被并发任务连续访问，而位于中等重用区域和低重用区域的索引则只向上升一级，因为位于这两个区域的索引的可重用性有限，逐步提升使得它们能够有机会被重用，同时又能不长时间地占用缓存空间。

（2.3）驱逐策略：当缓存空间已满时，最低缓存优先级的依赖索引或图数据将被优先替换出缓存。同时长时间未被命中的依赖索引将逐步降级，使得缓存管理能够更加灵活有效。

总体而言，通过本发明所构思的以上技术方案与现有技术相比，能够取得下列有益效果：

1、迭代图处理的收敛速度快：本发明通过依赖驱动的边预取，使得图顶点状态能够沿依赖路径进行高效传播，同时通过图顶点间的依赖关系转换进一步加速图顶点状态传播效率，从而有效提高了迭代图处理的收敛速度。

2、多核处理器的有效利用率高：本发明通过依赖驱动的边预取，使得处理核能够沿路径进行图处理，从而有效减少不必要的图顶点更新，并减少处理核间的同步开销。而图顶点间的依赖关系转换则使得多核处理器能够高并发地对多条路径进行处理，进一步促进了对多核处理器的有效利用。

**附图说明**

图3为本发明硬件加速器架构图。

图4为本发明预处理的流程图。

图5为本发明图处理方法的流程图。

**具体实施方式**

为了使本发明的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本发明进行进一步详细说明。应当理解，此处所描述的具体实施例仅用以解释本方面，并不用于限定本发明。此外，下面所描述的本发明的各个实施方式中所涉及到的技术特征只要彼此之间未构成冲突就可以相互组合。

本发明提出的高能效软硬协同图计算方法及装置，其主要思想为：每轮迭代时，与处理核耦合的硬件加速器沿依赖路径预取图数据，以供处理核进行访问处理，使图顶点状态能够沿依赖路径进行高效传播。同时，硬件加速器还维护了一组核心依赖路径首尾顶点之间的直接依赖关系，从而进一步加速图顶点状态传播，并最大限度地提高了多核处理器的有效并行度。

实施例所提供的高能效软硬协同图计算方法及装置，其流程如图4和图5所示，包括图数据预处理阶段和依赖驱动的异步图处理阶段，具体如下：

（1）图数据预处理。获取给定图的中心顶点与中心路径，并在此基础上划分核心子图、核心路径和核心顶点。包括如下子步骤：

（1.1）遍历图顶点，取大于度数阈值的顶点为中心顶点。度数阈值的计算方法为：根据用户指定的中心顶点的比例计算出中心顶点数量（为所有顶点数量），然后根据顶点度数对所有顶点进行降序排序，取第个顶点的度数为度数阈值。因对所有顶点进行排序的代价太高，所以也可以使用采样方法快速确定度数阈值，取占比为的采样顶点并对其降序排序，取第个顶点的度数为度数阈值。

（1.2）遍历图数据，基于中心顶点获取中心路径，即首尾顶点均为中心顶点的路径，中心路径的并集构成核心子图。具体而言，从步骤（1.1）中划分出的中心顶点集获取中心顶点，以此为根顶点执行深度优先搜索遍历，遍历过程中优先访问度数高的顶点，并指定遍历深度（默认深度为16），设遍历路径的首顶点为，尾顶点为，若属于中心顶点集，将顶点标记为已访问并将路径加入集合，若中心顶点集中的顶点均被标记为已访问则直接结束遍历。当前遍历完成后，若中心顶点集中的顶点未被全部访问，则选取中心顶点集中的下一个未访问顶点作为根顶点继续执行上述遍历，直至中心顶点集中的顶点均被访问，最终获得所有中心路径即为核心子图。

（1.3）遍历核心子图，将核心子图划分为相交顶点为路径首尾顶点的路径，称为核心路径。同时获取两条核心路径相交的顶点，称为核心顶点。具体而言，从步骤（1.2）中划分出的核心子图获取具有未访问边的顶点（优先选取中心顶点），从该顶点出发沿深度优先搜索遍历的顺序获取一条路径（路径最大长度默认为16），并将路径的首尾顶点加入核心顶点集合，重复上述步骤，直至核心子图的所有边都被访问。

（2）依赖驱动的异步图处理。通过硬件加速器的依赖路径预取单元从活跃顶点开始沿依赖路径预取边进行处理，同时通过硬件加速器的直接依赖管理单元将核心依赖路径首尾顶点间的依赖关系转换为直接依赖并对其进行缓存管理，在路径首顶点状态更新后，根据直接依赖公式快速计算出对路径尾顶点的影响并更新尾顶点状态。具体包括如下子步骤：

（2.1）硬件加速器初始化。通过调用加速器配置接口将图数据信息传递到加速器可访问的内存映射寄存器来初始化该加速器。默认传递的图数据信息包括：（a）CSR格式图数据所包括的偏移数组、边数组和顶点状态数组的大小和首地址；（b）分配给对应处理核的图分区的起始顶点与结束顶点；（c）图分区中核心顶点集合的大小和首地址；（d）对应处理核的本地循环队列的大小和首地址，该循环队列用于存储图分区中待处理的活跃顶点。

（2.2）依赖驱动的边预取。在每一轮图处理中，硬件加速器的依赖路径预取单元从本地循环队列中的活动顶点开始，通过深度优先搜索为其对应的处理核动态预取图分区的边。依赖路径预取单元使用一个固定深度的栈记录预取信息，栈中的每个条目都包含以下信息：（a）遍历中所访问顶点的；（b）该顶点的未访问边的当前偏移量与结束偏移量；（c）该顶点的未访问邻居顶点的。

硬件加速器的依赖路径预取单元以四段流水线形式完成边的预取，每次都将获取的边以及该边对应的一对顶点的状态输出到边缓冲区供处理核访问处理：

（2.2.1）若栈为空，则从本地循环队列中获取一个活动顶点，并将其压栈。

（2.2.2）从偏移数组获取栈顶顶点的出边起始/结束偏移。

（2.2.3）根据该顶点的未访问边获取未访问邻居顶点的，并将其中一个邻居顶点压栈。

（2.2.4）从顶点状态数组获取相关顶点的状态，将该边以及该边对应的一对顶点的状态输出到边缓冲区。若栈顶顶点属于顶点集合，则栈顶顶点出栈，并将其插入本地循环队列作为新的活跃顶点，进入步骤（2.2.1）。若无法从栈顶顶点的邻居中获取到图分区中的未访问顶点，则栈顶顶点出栈，进入步骤（2.2.1）。

（2.3）图顶点依赖关系转换。在依赖路径预取单元进行边预取的同时，直接依赖管理单元将核心依赖路径首尾顶点的间接依赖转换为直接依赖。当顶点之间的依赖关系为线性关系时，两个顶点的直接依赖关系可以表示为公式，其中、为顶点、的状态值，、为常数参数。直接依赖管理单元使用直接依赖索引数组保存路径首尾顶点间的直接依赖索引，数组中的每一项索引都包括首顶点编号、尾顶点编号、路径标识、参数、参数和索引标识。其中索引标识代表该索引的当前状态，一共分为三种情况：（a）索引标识为，则该索引无效；（b）索引标识为，则参数、参数的当前值为顶点、的一组状态值、；（c）索引标识为，则索引有效，参数、参数的值为直接依赖公式的参数值。依赖转换过程如下：

（2.3.1）直接依赖索引的索引标识初始化为，图处理过程中第一次完成对核心依赖路径（首尾顶点都属于顶点集的路径）的处理时，将其首尾顶点的编号、以及状态值、保存至直接依赖索引数组，并将索引标识设置为。

（2.3.2）第二次完成对路径的处理时，获得首尾顶点一组新的状态值，结合已存储在该索引、处的上一组状态值，连立代入直接依赖公式计算出常数参数、的值，将和的值保存至索引、处，并将索引标识为设置为。

在后续的处理中，对路径首顶点进行更新后可以通过直接依赖索引获取直接依赖公式的参数，然后根据公式计算出首顶点更新对尾顶点的影响并更新尾顶点状态。此外，直接依赖管理单元使用定制化缓存策略对依赖索引进行缓存管理：

（1）依赖索引的可重用性划分。首先根据依赖源顶点的度数降序排列所有依赖索引，排列开始处的一个LLC大小的区域被划分为高重用区域，在高重用区域后的一个LLC大小的区域被划分为中等重用区域，剩余区域则被划分为低重用区域，每个区域中的依赖索引拥有对应等级的可重用性。

（2）定制化缓存策略包括插入策略、命中提升策略以及驱逐策略，具体如下：

（2.1）插入策略：当被访问的依赖索引未被缓存时，则将索引插入缓存，并根据索引所处的可重用区域设置不同的缓存优先级，否则不插入该索引。高重用区域的索引被设置为高优先级，避免被马上替换出缓存；中等重用区域的索引被设置为中等优先级，使得中等访问频率的索引在不导致缓存抖动的情况下也能有机会被重用；而低重用区域的索引与图数据则被设置为低优先级，使得它们能够立即被替换出缓存，从而更有效地利用缓存空间。

（2.2）命中提升策略：当依赖索引被命中时，升级该索引的缓存优先级。高重用区域的索引被立即升级为最高优先级，因为它们接下来很可能会被并发任务连续访问，而位于中等重用区域和低重用区域的索引则只向上升一级，因为位于这两个区域的索引的可重用性有限，逐步提升使得它们能够有机会被重用，同时又能不长时间地占用缓存空间。

（2.3）驱逐策略：当缓存空间已满时，最低缓存优先级的依赖索引或图数据将被优先替换出缓存。同时长时间未被命中的依赖索引将逐步降级，使得缓存管理能够更加灵活有效。

本专利对于满足权利要求5所述两点属性的图算法均表现出良好的性能，大多数迭代图算法都满足这两点属性，如pagerank、adsorption、SSSP、WCC、k-core等，下面以SSSP算法为例详细描述本发明的实施过程：首先遍历图顶点，取大于度数阈值的顶点为中心顶点，然后基于中心顶点遍历图数据以获取中心路径，即首尾顶点均为中心顶点的路径，从而得到由中心路径并集构成的核心子图，再遍历核心子图，将核心子图划分为相交顶点为路径首尾顶点的核心路径，同时获取两条核心路径相交的顶点，即核心顶点。完成图数据预处理后，通过调用加速器配置接口，将图数据信息传递到加速器可访问的内存映射寄存器来初始化硬件加速器。在每一轮图处理中，硬件加速器的依赖路径预取单元从本地循环队列中的活动顶点开始，通过深度优先搜索为其对应的处理核动态预取对应图分区的边。在依赖路径预取单元进行边预取的同时，直接依赖管理单元将核心依赖路径首尾顶点的间接依赖转换为直接依赖并对其进行缓存管理。在SSSP算法中，两个顶点的直接依赖关系可以表示为公式，其中、为顶点、的状态值，、为常数参数(SSSP算法中，参数恒为1)。在首次处理核心依赖路径时，保存路径首尾顶点的第一组顶点状态值（，），在第二次处理路径时，保存第二组顶点状态值（），然后代入直接依赖公式即可计算出参数、的值。在后续的处理中，对路径的首顶点进行更新后，可以通过直接依赖索引获取直接依赖公式的参数，然后根据公式计算出首顶点更新对尾顶点的影响并更新尾顶点状态，而不需要等待路径首顶点状态经过多轮迭代传播到尾顶点，从而可以在多个核上高并发地对多条路径进行处理，加速图顶点状态的传播，提高图计算的收敛速度。

本领域的技术人员容易理解，以上所述仅为本发明的较佳实施例而已，并不用以限制本发明，凡在本发明的精神和原则之内所作任何修改，等同替换和改进等，均应包含在本发明的保护范围之内。

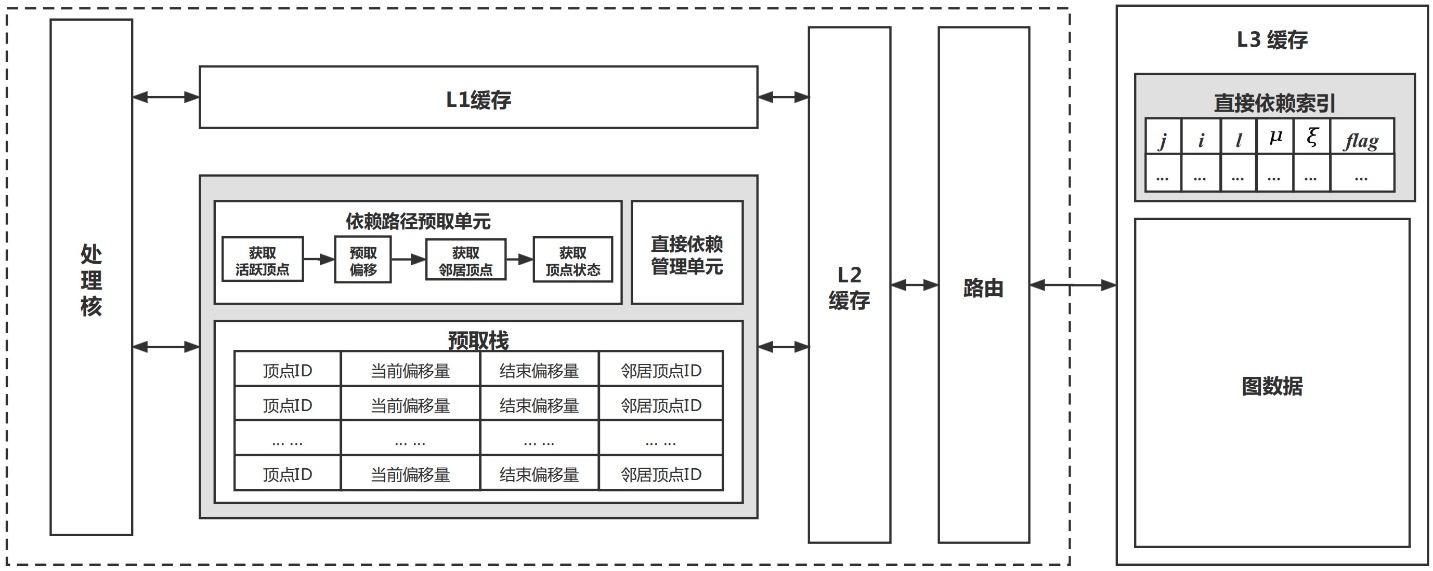
**说 明 书 附 图**

图3 硬件加速器架构

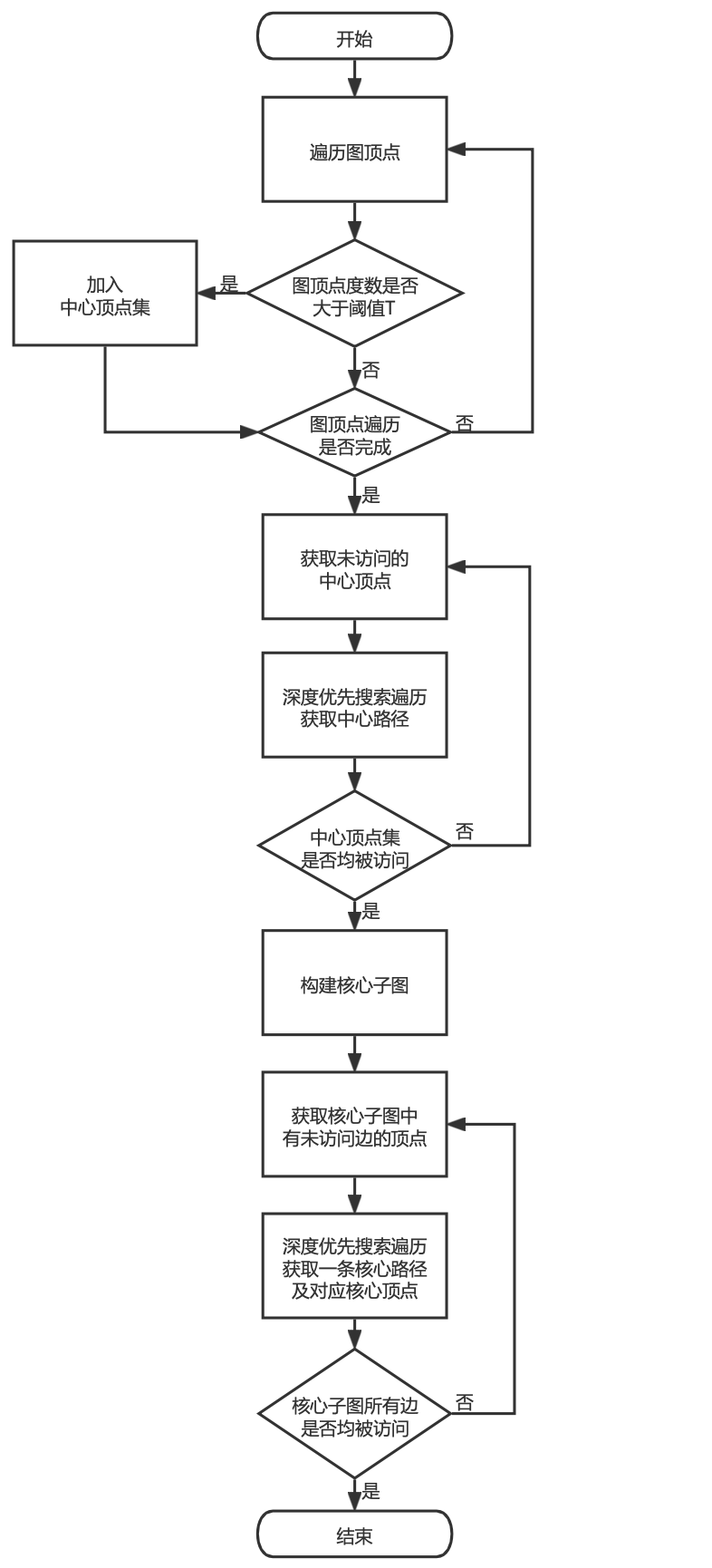


图4 预处理流程图

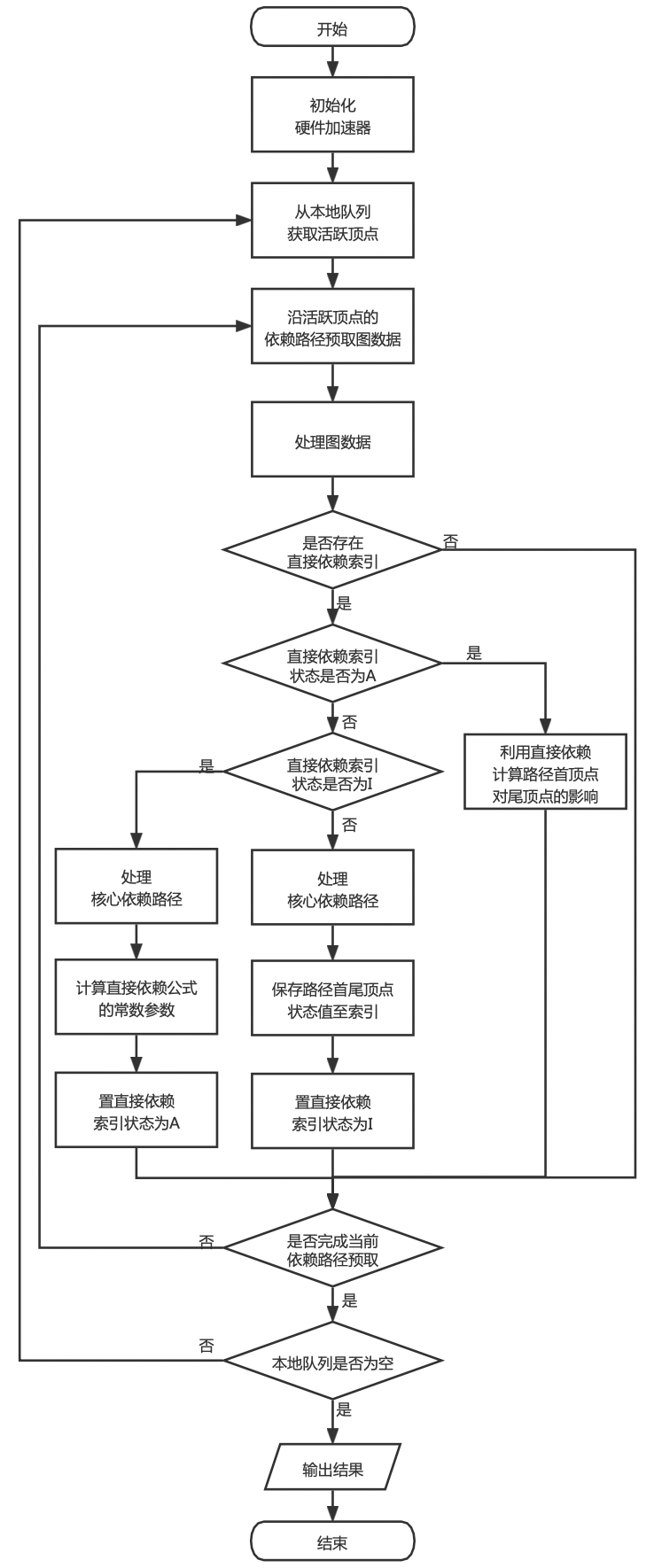


图5 图处理流程图