基于路由机制的时变路网 k 近邻算法

张栋良[1] 唐 俊[2]
(上海电力学院电力系自动化工程学院 上海 200090)[1]
(同济大学嵌入式系统与服务计算教育部重点实验室 上海 201804)[2]

摘 要 针对现实生活中动态路网的地理信息查询问题,提出了一种基于路由机制的动态路网中 k 近邻查询的算法。其主导思想是利用空间换时间,用路由表保存历史查询结果,用查询路由表的方法代替传统的最短路径计算,通过历史数据减少系统重复计算并对车辆行驶路径进行规划,用更新路由表的方法适应路况的变化。围绕路由表这一核心,改进相应的 k 近邻算法的过滤、精炼过程。通过路由表对动态路网进行少量的预处理,减少系统在 k 近邻搜索中的候选点数量,缩小查询范围,提高搜索效率。

关键词 路由机制,k 近邻算法,时变路网
中图法分类号 TP391.9 **文献标识码** A

k-Nearest Neighbor Algorithm in Dynamic Road Network Based on Routing Mechanism

ZHANG Dong-liang[1] TANG Jun[2]
(Department of Electric Power and Automation Engineering, Shanghai University of Electric Power, Shanghai 200090, China)[1]
(Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai 201804, China)[2]

Abstract Aiming at the issue of geography information query, a new k-Nearest Neighbor algorithm for dynamic road network was proposed based on routing mechanism. With the idea of "space for time", we saved history query results in routing tables, and substituted the traditional method by requiring tables. We updated the route tables to adapt the time varying road status. With the kernel of routing table, we improved the filtering and refining procedure of kNN algorithm. By preprocess of dynamic road network using routing table, the amount of candidate points in k-NN computing is reduced, and the rang of query and the searching efficiency are promoted.

Keywords Routing mechanism, k-nearest neighbor algorithm, Dynamic road network

1 引言

随着普适计算技术、无线通信技术以及全球定位技术的发展,高效的位置相关查询及基于位置的信息服务(LBS)得到了人们日益广泛的重视[1],k 近邻服务是其中研究热点之一。k 近邻服务作为 LBS 应用中最具发展潜力的研究方向之一,在车辆导航、在线地图服务、公共交通预测到达等方面都有着巨大的研究空间。

拓扑网络中的 k 近邻搜索问题不同于散乱点中的 k 近邻搜索[2]。其搜索对象的位置和运动被约束在拓扑网络上,因此搜索 k 近邻时需要计算其网络距离[3],而不能简单地计算对象之间的欧氏距离。近几年来,国内外在搜索静态路网中的 k 近邻对象并同时求出点到这些 k 近邻对象的路径方面已经展开了相关的研究,但针对动态路网(即时变路网,Time-Dependent Road Network)中 k 近邻搜索的研究还很有限,国外已有相关的一些研究和应用,而在国内仅有很少量的研究和应用,即处于起步阶段。

基于 R 树的经典 k 近邻算法由 Roussopoulus、Kelley 和 Vincent 于 1995 年提出的深度优先(Depth First, DF)算法[4]改进而来。这种方法将 Mindist 和 MinMaxdist 作为查询判断条件,深度优先遍历 R 树查询最近邻。这种方法适用于静态环境下的最近邻查询。Bespamyatnikh 和 Snoeyink 提出了一种静态环境下基于 Voronoi 图的移动查询点最近邻查询算法[5],它能求得查询点的一个最近邻。郝忠孝在此基础上提出一种基于一阶 Voronoi 图的静态环境下 k 近邻查询算法[6]。该算法利用一阶 Voronoi 图的性质把 k 最近邻查询的查询空间限定在一个特定的区域,降低了查询的时间复杂度。

近年来,由于移动设备的广泛应用,面向路网中的移动对象 k 近邻查询受到很多研究者的关注。文献[7-9]中提出的是面向静态路网上的移动对象(如出租车)查询的 k 近邻算法,其特点是查询对象的动态性,但路网上的路况是不变的,而本文提出的方法是针对路况时时发生变化的动态路网的 k 近邻算法,更具有实用性。

Demiryurek 等人针对动态路网中道路权值随着时间而不断变化的问题,提出一种动态路网上的 k 近邻算法[10],它使用 TNI(Tight Network Index)索引和 LNI(Loose Network

到稿日期:2012-05-01 返修日期:2012-09-12 本文受国家自然科学基金青年基金项目(61003089)资助。
张栋良(1977—),男,博士,讲师,主要研究方向为分布式计算、智能交通,E-mail:fire_zdl@163.com;唐 俊(1987—),男,硕士,主要研究方向为移动计算与智能交通。

Index)索引，在仅作少量预处理的前提下，尽可能地减少候选点数，缩小最短路的计算开销。但计算环节还是采取了传统的计算方式，本文在计算环节上采取的是计算与查询相结合的方式。

本文针对现实生活中的动态路网，并结合国内外的现有研究成果，提出了一种基于路由机制的动态路网中最短路径的求解方法[11]。主要思想是利用路由表保存过去一段时间内的历史查询结果，通过历史数据减少系统重复计算并对车辆行驶路径进行规划。本文以路由表为核心，改进 k 近邻算法的过滤、精炼等过程，通过路由表对动态路网进行少量的预处理，减少系统在 k 近邻搜索中的候选点数量，缩小查询范围，提高搜索效率。

2 路网对象建模

路网中的查询对象数量往往数以百万计，因此一般采用索引结构进行组织，以节省查询时间。由于对象在路网中的坐标位置固定，因此其位置也可抽象成路网中的一个位置或结点。而且每一种类型的对象的数量较少，通常只有几百到几千的数量规模。

定义 1 时变路网（Time-dependent Road Network，TDRN）为 $G_T(V,E)$。其中 V 代表道路结点，E 代表道路结点之间的路段。对于任意路段 $e(v_i,v_j)\in E$，其路段权值为 $c(v_i,v_j,t)$，代表车辆在时刻 t 从 v_i 到 v_j 的行驶代价。时变路网在文中也称为动态路网 G_D。

定义 2 时变路网 $G_T(V,E)$ 上的对象 q_I 可表示为 5 元组（id，$name$，loc，$type$，$level$，$keywordList$）。其中 id 为唯一标识，其值唯一；$name$ 为其的实例名称；loc 标识 q_I 在路网上的具体位置；$type$ 为其类别，例如医院、商店等；$keywordList$ 为关键字表，关键字 $keyword$ 具体描述 $type$，例如三级甲等、华山、皮肤科等。对象在本文中也称为兴趣点（Point of Interest，POI）。

定义 3 最小边权路网 $G_M(V,E)$ 真包含于时变路网 $G_T(V,E)$，且对于任意路段 $e(v_i,v_j)\in E$ 及任意时间 t，其权值 $c(v_i,v_j)\in G_M=\min c(v_i,v_j,t)\in G_T$，最小边权路网在本文中也称为静态路网 G_S。

定义 4 地图的非叶结点索引 I_{index} 是一棵 5 元组的 PR-QUAD 树〈$Rect$，$*ul$，$*ur$，$*dl$，$*dr$〉。其中 $Rect$ 标记该索引的区域范围，$*ul$、$*ur$、$*dl$、$*dr$ 是指向其子树或叶子结点的指针。叶子结点索引 I_{leaf} 是一个 3 元组〈$Rect$，$list$:〈q_I*〉，$list$:〈q_M*〉〉。其中 $Rect$ 标记 I_{leaf} 索引的范围，$list$:〈q_I*〉和 $list$:〈q_M*〉分别指向索引区域中兴趣点的指针列表，列表的大小不超过设定的阈值大小。

定义 5 路由表 T 为三元组〈$curId$，$destId$，$nextId$〉，其中 $curId$ 为当前节点 id，$destId$ 为目的地节点 id，$nextId$ 表示车辆从 $curId$ 开往 $destId$ 时，其下一步应往 $nextId$ 结点行驶。车辆在每到达一个道路结点时便访问路由表来获取下一步的行驶方向，直到行驶到目的地。

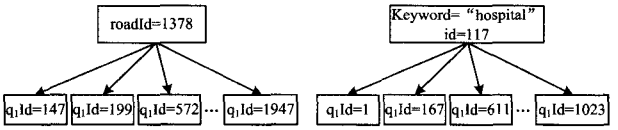
3 查询对象的索引和查询

3.1 查询对象的索引表结构

查询对象的索引分为正向索引和反向索引，如图 1 所示。正向索引为道路路段映射到对象上的索引，即给定一路段，找到在该路段上的所有 q_I，通过 q_I 的 id 值查询其名称、类别、关键字、属性等信息。反向索引是针对某一关键字而索引的对象列表。当用户给出查询请求时，通过反向索引可快速获得初始候选集。

定义 6 正向索引表为二元组（$roadId$，$objIdList$）。其中 $roadId$ 为道路结点 id 号，$objIdList$ 为兴趣点的有序列表，表示为 $\{objId_1,\cdots,objId_i,\cdots,objId_n\}$，$1\leqslant i\leqslant n$。$objId_i$ 标识兴趣点 id，且对于任意 $i\geqslant 2$，有 $objID_{i-1}\leqslant objID_i$。

定义 7 反向索引表为二元组（$keywordId$，$objIdList$）。其中 $keywordId$ 为关键字 id 号，$objIdList$ 为兴趣点的有序列表。



(a) 正向索引表　(b) 反向索引表

图 1

3.2 候选集的筛选

路由机制在动态路网 k 近邻算法中的作用包括候选结点集的筛选、路网搜索区域的裁剪等。

定义 8 动态路网 $G_D(V,E)$ 对应产生的最小边权路网为静态路网 $G_S(V,E)$，G_S 为 G_D 的最小边权路网。即对于任意路段 $e(v_i,v_j)\in G_S$，其路段权值 $c(v_i,v_j)\leqslant c(v_i,v_j,t)\in G_D$。

在静态路网 G_S 上，用最短路算法对其任意两点间的最短路径进行与计算，且将计算所得的最短路径树分解并保存到路由表[8]，得到静态路由表 T_S。显然，相对于动态路网 G_D 上的路由表 T_D，T_S 中任意两点间行驶路径的代价和总低于 T_D 上的。

定理 1 静态路网 G_S 上任意两点之间的最短距离 d_s 总要小于或者等于 G_S 对应的动态路网 G_D 上该两点之间的最短距离 d_d。

证明：反证法。若定理不成立，假设对于选定的两点 v_i，v_j，有 $d_s(v_i,v_j)>d_d(v_i,v_j)$。由定义 8 可知，在 G_S 中构造与路径 $d_d:v_i\rightarrow v_j$ 相同走向的新路径 $d_{s'}:v_i\rightarrow v_j$，满足 $d_{s'}\leqslant d_d$。用该路径 $d_{s'}:v_i\rightarrow v_j$ 替换 G_S 中的现有路径 $d_s:v_i\rightarrow v_j$ 可以得到一条比原先代价更小的路径。这显然与 d_s 是 G_S 中的最短距离相矛盾。得证。

定理 2 设静态路网 G_S 上的路由表为 T_S，G_S 对应的动态路网 G_D 上的路由表为 T_D。则 T_S 上任意两点间的指路距离总和 d_{ts} 总要小于或者等于该两点在 T_D 上的指路距离总和 d_{td}。

证明：因为 G_S 中的路网权值不变化，可知对于任意选定的两点 v_i，v_j，$d_{ts}:v_i\rightarrow v_j$ 代表静态路由表 T_S 中两点间的行驶路径，$d_{Gs}:v_i\rightarrow v_j$ 代表 G_S 中两点间的最短行驶路径，有 $d_{Ts}:v_i\rightarrow v_j\equiv d_{Gs}:v_i\rightarrow v_j$ 且行驶代价 $d_{Ts}=d_{Gs}$。设 G_D 中两点间的最短行驶路径为 $d_{Gd}:v_i\rightarrow v_j$，其对应的路由表 T_D 中两点间的行驶路径为 $d_{Td}:v_i\rightarrow v_j$，有 $d_{Gd}\leqslant d_{Td}$。由定理 1 可知，$d_{Gs}\leqslant d_{Gd}$。因此有 $d_{Ts}=d_{Gs}\leqslant d_{Gd}\leqslant d_{Td}$。得证。

根据定理 1 及定理 2，在给定路由表 T_S、T_D 及候选点点集 P 的基础上，通过对比候选点的实际距离和最小距离，可

过滤部分候选点。其筛除依据是：若某候选点 p_n 到查询点的最小距离大于所有候选点中估算距离为第 k 近的候选点 p_k，则 p_n 必定不在 k 近邻查询的结果中。

候选点筛选的算法伪代码见图 2。算法第 2－4 行根据路由表 T_D 内已保存的路径，计算车辆结点到每个候选点的行驶路径总开销作为估算距离。第 5 行根据 q 到各候选点的距离由近到远将前 k 个近邻候选点保存到候选点集 Q 中。第 6－12 行用静态路由表 T_S 内的指路信息计算车辆到各剩余候选点的理论最小代价，过滤候选集。

算法 1 select-kNN-Candidate(P,q,k,t)

输入：初始候选集 P，查询点 q，查询数 k，当前时刻 t

输出：过滤后的候选集 Q

1. Q←φ，DD←φ　// DD 保存 q 到各候选点的距离
2. for p∈P (location near to q by visiting I_{leaf} and I_{index}) do
3. 　DD←d_p←dist-in-T_D(q,p,t)　// 在动态路由表 T_D 中计算距离
4. end for
5. Q←{$p_1,\cdots,p_k$} ascend by d_{pi} in DD
6. kth_estimate_dist=maximum d(q,q_i) (q_i∈Q,1≤i≤k)
7. for p_i∈P−Q do
8. 　d_s=dist-in-T_S(q,p_i)　// 通过 T_S 计算最小可能距离
9. 　if d_s≤kth_estimate_dist do
10. 　　Q←p_i
11. 　end if
12. end for
13. Return Q

图 2　候选集的筛选算法

定理 3　候选集的筛选算法筛除的点必定不是 k 近邻中的任意一个。

证明：假设候选点的数量为 k_c（$k_c>k$）。通过动态路网 G_D 对应的路由表 T_D，对这些候选点进行距离估算并排序。排在前 k 近邻的 k 个点仍作为有效候选点进行保留。取前 k 个有效候选点中估算距离的最大值作为筛选阈值 d_{kth}。对于任意剩余候选点 $p_r\in P$，用静态路由表 T_S 内的数据计算其理论最小值 d_s。若 $d_s>d_{kth}$，通过定理 2 可知，无论如何 p_r 都不会是前 k 个候选点中的一个。证毕。

假设路网上的道路结点数为 n，车辆行驶路径中经过的平均结点数为 l，查询量为 k，P 中元素总数为 m，其中 $k<<m$，$l<<n$。则在动态路由表 T_D 为空时，算法 1 的时间复杂度为：

$$C_1=m\mathrm{O}(n\lg n)+\mathrm{O}(m)+(m-k)l=\mathrm{O}(mn\lg n) \quad (1)$$

式中，$\mathrm{O}(n\lg n)$ 为基于路由机制的路径快速生成算法在计算路径并将最短路径树保存至路由表时的时间复杂度。$\mathrm{O}(m)$ 为从 P 中获取前 k 个元素的时间复杂度。$(m-k)l$ 为在静态路由表 T_S 中计算查询点 q 到剩余点的最小可能距离时的计算开销。

然而当动态路由表 T_D 中已有大量历史数据时，算法在第 2－4 行时可以通过简单的查表获得 q 到候选点的估算距离，时间开销为 ml。因此时间复杂度降到：

$$C_2=ml+\mathrm{O}(m)+(m-k)l=\mathrm{O}(m)+(2m-k)l=\mathrm{O}(m) \quad (2)$$

由此可知，当动态路由表中缺少历史数据时，候选点的筛选算法需要大量计算两点间的最短路径并更新至动态路由表中，时间复杂度由最短路算法决定，性能相对较差；而在路由表中已有大量历史数据的前提下，时间复杂度仅由候选集的大小决定，算法性能大幅提高，其时间开销大幅降低。

3.3　路网区域的裁剪

根据静态路网 G_S 上的路径信息，从查询点 q 的起点 $startV$ 到最近的 k 个兴趣点的完整行驶路径，以该行驶路径中各路段的最大行驶代价代替 G_S 中的路段权值，计算最大可能行驶代价值 $d_{\max}$。即对于任意 OD 点（$v_i=startV,v_j=q_l,1\leqslant j\leqslant k$），有行驶路径 $v_i\rightarrow v_j$。则对于任意路段 $e(v_x,v_{x+1})$，$i\leqslant x<j$，$d_{\max}=\sum\max\, c(v_x,v_{x+1},t)$。以 $d_{\max}$ 为阈值，若地图上一点 p，使得 q 到 p 的最小行驶代价超过 $d_{\max}$，则任意 k 近邻对象不会在点 p 上，且 q 到任意 k 近邻的行驶路径不会经过点 p。该点将被裁剪。$d_{\max}$ 为 k 近邻的距离上界。

定理 4　设查询点 q 到 k 近邻的最大可能代价为 $d_{\max}$。对地图上的任意点 p_m，若 q 到 p_m 的最小可能代价 $d>d_{\max}$，则前 k 个近邻不在点 p_m 的位置上，且 q 到前 k 近邻的行驶路径不经过点 p_m。

证明：反证法。假设前 k 个近邻中存在点 p，其在 p_m 位置上。设 q 所在道路结点为 v_1，p 所在道路结点为 v_m，则其行驶路径为 $v_i\rightarrow v_m$，行驶代价 $d(q,p)>d_{\max}$。根据阈值 $d_{\max}$ 的定义可知，$d_{\max}\geqslant d(q,p_i)$，$1\leqslant i\leqslant k$，即有 $d_{\max}\geqslant d(q,p)$，与 $d(q,p)>d_{\max}$ 相矛盾。而若存在 q 到前 k 近邻的一条行驶路径经过点 p，则假设 p_r 为 q 的第 r 个近邻，其中 $1\leqslant r\leqslant k$，p_r 所在道路结点为 v_r。则其行驶路径 $v_1\rightarrow v_r=v_1\rightarrow v_m+v_m\rightarrow v_r$。因此 $d(q,p_r)=d(q,p_m)+d(p_m,p_r)>d_{\max}$。与 $d_{\max}>d(q,p_r)$ 矛盾。证毕。

在原始地图 $G(V,E)$ 下，通过静态路由表 T_S 来获得第 k 近邻的最大可能距离 $d_{\max}$。然后根据 T_S 表内信息，计算查询点到地图 G 中任意结点之间的最小可能距离 $dist$。若 $dist>d_{\max}$，根据定理 3，可将该点及从起始点出发且中间经过该点的其他所有结点一并裁减掉。则最终经裁剪后的地图 $G_E(V_E,E_E)$ 中，V_E 为结点集 Q_2，$E_E=\{edge(v_i,v_j)\mid edge(v_i,v_j)\in E,v_i,v_j\in Q_2\}$。

假设路网上的道路结点数为 n，平均每个 PR-QUAD 树的索引叶结点区域包含 m 个道路结点，遍历到的 PR-QUAD 树索引叶结点有 j 个，车辆行驶路径中经过的平均结点数为 l，其中 $m<<n$，$l<<n$，则路网裁剪的算法复杂度为：

$$C_3=kl+j(l+m)+\mathrm{O}(m)=\mathrm{O}(m) \quad (3)$$

4　对象的 k 近邻查询

通过对路由机制的适当调整，便可使得路由表适应动态路网中的 k 近邻查询操作。路网中的道路权值变化一般依赖于车流量及出行规模。对于各路段 $e(v_i,v_j)\in E$ 的道路权值变化，其有极强的规律性。例如以天为统计单位，则路段权值变化一般如图 3 所示，每天的变化曲线趋近一致，并且不同路段的曲线也相似。

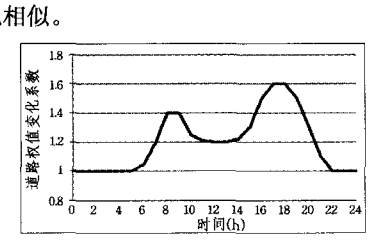


图 3　一天内的某路段道路权值变化曲线

算法 2 calc-realtime-route(G,T,q,k,t)

输入:地图 G,T 对象类型,查询点 q,查询数 k,当前时刻 t

输出:容量为 k 的有序集 P_k

1. Q_1←check-index(T)
2. Q_2←select-kNN-Candidate(Q_1,q,k,t)
3. G′←cut-map(G,Q_2,q)
4. for p∈Q_2 do
5. P_k←⟨p,dist⟩←dist←d(q,p,t) by fast-path-generation-algorithm in G′
6. end for
7. return P_k

图 4 对象的 k 近邻查询算法

因此可根据路网上各路段的道路权值变化曲线,选择一定的权值变化系数阈值 $threshold_1$ 与路由表的清空阈值 $threshold_2$。当路网中有部分路段权值变化 $threshold_1$ 时,用路由表的动态更新策略来更新路网上的指路信息;当有超过 $threshold_2$ 数量的道路权值均发生改变时,清空旧的动态路由表,即清空历史数据,对新的查询请求重新计算并保存至动态路由表中,使得路由表内的信息始终符合路网权值的变化情况。

对象的 k 近邻查询思想为:提取并解析用户请求数据后,通过反向索引表查找出初始候选集 Q,通过正向索引表比对 $type$ 等信息过滤无效对象后形成候选集 Q_1。然后通过算法 1 筛选候选集得到过滤后的候选集 Q_2。最后利用路径计算得出最终的结果并返回用户。

图 4 描述了对象的 k 近邻查询算法。P_k 是容量为 k 的有序集,按照查询点 q 到候选点的实时代价由小到大排序,优先删除最大值。算法 2 的第 4—6 行一次性调用扩展的路径快速生成算法[8]搜索候选集中的实时路径。最后返回查询点 q 的前 k 近邻。

假设路网中的道路结点数为 n,候选集中的元素个数为 $m(m<<n)$。则在路由表中已有一定历史数据的前提下,对对象的 k 近邻查询复杂度为式(4)。

$$C_4=O(\lg n)+O(m)+O(m)+O(n\lg n)=O(n\lg n) \quad (4)$$

本文提出的 k 近邻算法及路径快速生成算法均基于路由机制。而路由机制的实现,是在最短路计算的基础上,通过路由表分解并保存已计算的最短路径树所附带的结点前趋/后继关系作为历史数据,从而在理论意义上减少系统的重复计算,缩短系统的相应时间。因此针对基于路由机制的 k 近邻算法,提出:

1)在动态路由表的一个更新周期内,大规模仿真车辆并发 k 近邻查询和导航查询时减少的查询次数。

2)单个动态路由表的更新与两个路由表协作更新策略的时间对比。

3)不同的动态路由表重置间隔对实验系统在响应请求时间方面的影响。

基于路由机制的 k 近邻算法需要对静态路由表 T_S 进行预处理。实验中,在该地图上的预处理时间约为 35 秒。我们定义同一时刻的 10000 次并发查询为 1 次测试。

第一项测试共进行连续的 20 次测试,分别统计导航查询与 k 近邻查询在每次测试时减少的重复计算数。在一个周期(即测试周期内不重置并更换动态路由表 T_D)内的实验结果如图 5 所示,本文提出的路由机制在 k 近邻算法及导航查询时的最短路算法上均能有效地减少重复计算。在导航查询测试的中后期,减少的重复计算数接近 10000,这是因为随着大量的导航查询请求和计算所得的最短路径树的分解与存储,路由表中已保存有足够多的历史数据。这些历史数据的存在避免了大量的重复计算。在 k 近邻查询测试中,k 通常的取值范围为[2,5],使得对于任意 k 近邻查询请求,路由表中已存在全部解的概率相对导航查询请求低,因此对比数据中其减少的重复计算量要低于前者。

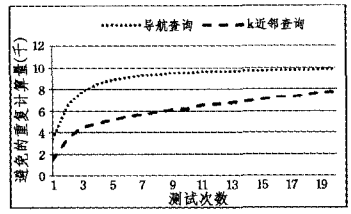


图 5 k 近邻查询和导航查询时减少的重复计算次数

第二项测试中,测试单动态路由表与双动态路由表协作更新策略之间的性能对比。共分为 2 组,每组 20 次测试。其中,路由表的重置时刻 t_0 为第 11 次测试的起始时刻,协作更新的起始时刻 t_1 为第 9 次测试的起始时刻,即阈值 $\Delta t=t_0-t_1$ 为 2 次测试的时间间隔。

实验结果如图 6 所示,2 种更新策略在前 8 次的实验数据基本一致。双动态路由表的协作更新策略在第 9—10 次的测试里要比单动态路由表多花几秒的时间。这是因为在这 2 次测试里,双动态路由表的更新策略中需要将计算结果更新到 2 个路由表中,需要多花一定的时间,这在相对低负载的情况下是可接受的。在第 11 次系统清空旧的动态路由表后的测试中,使用双动态路由表协作更新策略的系统由于动态路由表在第 9—10 次测试时已保存了部分历史数据,因此总体上比单动态路由表的系统具有更好的性能。系统使用双动态路由表的协作更新策略,相对地减少了每次重置路由表后的计算量。显然,当阈值 $\Delta t\approx 0$ 时,双动态路由表协作更新策略退化为单动态路由表更新策略。

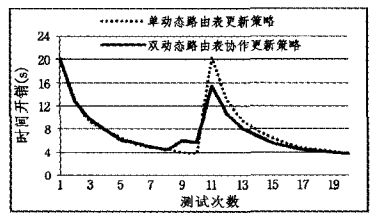


图 6 系统单、双动态路由表更新策略的时间开销

在第三项测试中,我们测试双动态路由表协作更新策略时不同的动态路由表重置间隔对系统性能的影响情况。共分为 2 组,每组 20 次测试(见图 7)。

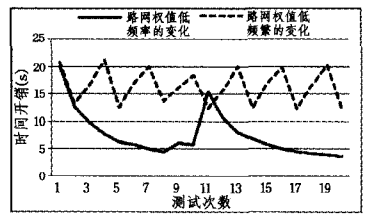


图 7 路网权值的频繁变化导致性能恶化

图 7 中标实线的时间开销曲线是路网权值变化较少导致的动态路由表的重置间隔较长的测试结果。该组测试中，路由表的重置时刻 t_0 为第 11 次测试的起始时刻，协作更新的起始时刻 t_1 为第 9 次测试的起始时刻，即阈值 $\Delta t = t_0 - t_1$ 为 2 次测试的时间间隔。

图 7 中标虚线的时间开销曲线是路网权值变化较频繁而导致动态路由表的充值间隔较短的测试结果。该组测试中，路由表的重置时刻 t_0 分别为第 4、7、10、13、16、19 次测试的起始时刻，协作更新的起始时刻为每次重置时刻的前一次测试起始时刻，即阈值 $\Delta t = 1$ 次测试的时间间隔。

结束语 基于路由机制的 k 近邻算法在实际应用时需要一定的前提条件，即在动态路网道路权值的变化不太频繁和剧烈的情况下，基于路由机制的 k 近邻算法既能通过实时计算保证结果的精确性，又能避免大量重复计算，显著降低计算开销。在路网权值变化较少，使得动态路由表的重置间隔较长时，随着动态路由表 T_D 内历史数据的逐渐丰富，计算时的时间开销显著降低；而当路网权值频繁变化而导致动态路由表的频繁重置时，因动态路由表始终无法存储足够多的历史数据而导致在候选集的筛选等步骤上的时间复杂度增加，并且无法大量地减少重复计算从而抵消或者降低路由机制中更新策略、导航策略等带来的额外的时间开销，导致性能降低。

参 考 文 献

[1] 孟小峰，丁治明. 移动数据管理概念与技术[M]. 北京：清华大学出版社，2009

[2] Goodsell G. On finding p-th nearest neighbors of scattered points in two dimensions for small p[J]. Computer Aided Geometric Design，2000，17：387-392

[3] Jensen C S，Pedersen K T B，et al. Nearest neighbor queries in road networks[C]// Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems. 2003：1-8

[4] Roussopoulos N，Kelley S，Vincent F. Nearest neighbor queries [C]// Proceedings of the ACM SIGMOD International Conference on Management of Data. 1995：71-79

[5] Bespamyatnikh S，Snoeyink J. Queries with segments in Voronoi diagrams[C]// SODA '99 Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms. 1999：122-129

[6] 郝忠孝. 时空数据库查询与推理[M]. 北京：科学出版社，2010

[7] 赵亮，陈荦，景宁，等. 道路网中的移动对象连续 K 近邻查询[J]. 计算机学报，2010，33(8)：1396-1404

[8] 陈子军，任彩平，刘文远. 路网中查询点速度不确定的连续 k 近邻查询方法[J]. 小型微型计算机系统，2011，32(3)：430-434

[9] 卢秉亮，刘娜. 路网中移动对象快照 K 近邻查询处理[J]. 计算机应用，2011，31(11)：3078-3083

[10] Demiryurek U，Banaei-Kashani F，Shahabi C. Efficient K-Nearest Neighbor Search in Time-Dependent Spatial Networks[C]// DEXA 2010，Part I，LNCS 6261. 2010：432-449

[11] 唐俊，张栋良. 基于路由机制的变权网络路径快速生成算法[J]. 计算机科学，2011，38(12)：110-124

(上接第 23 页)

结束语 为了在 FPGA 硬件空间上管理空闲资源全集 MFR，本文将栈技术引入到对 MFR 的查找与生成中，提出了一种利用单向栈的算法。本算法是 FPGA 上硬件任务实时在线放置与调度的基础，只有有效地管理好空闲资源全集，才能有效地应用任务调度算法。下一步我们将该基于单向栈的 MFR 全集管理算法应用于 CPU/FPGA 体系结构的集群系统上任务调度算法的研究。

参 考 文 献

[1] 余国良，伍卫国，杨志华，等. 一种采用边界表进行可重构资源管理及硬件任务调度的算法[J]. 计算机研究与发展，2011，48(4)：699-708

[2] 龚育昌，齐骥，胡楠，等. 部分可重构系统布局的一种新算法[J]. 中国科学技术大学学报，2007，37(9)：1047-1053

[3] 黄勋章，周学功，彭澄廉. 可重构系统中高效的二维任务放置策略[J]. 计算机工程与设计，2008，29(7)：1745-1749

[4] 焦铭，李仁发，李浪，等. 可重构系统中基于空间邻接度的实时任务放置算法[J]. 计算机应用研究，2011，28(4)：1290-1295

[5] Bazargan K，Kastner R，Sarrafzadeh M. Fast Template Placement For Reconfigurable Computing Systems[J]. IEEE Design and Test of Computers，2000，17(1)：68-83

[6] Walder H，Steiger C，Platzner M. Fast Online Task Placement on FPGAs：Free Space Partitioning and 2D hashing[C]// Proceedings of the 17th International Parallel and Distributed Processing Symposium. Washington D C，USA：IEEE Computer Society，2003：178

[7] Tabero J，Steptien J，Mecha H，et al. A Vertex-list Approach to 2D HW Multitasking Management in RTR FPGAs[C]// Proc. of DCIS. Cludad Real，Spain：[s. n.]，2003：545-550

[8] Handa M，Vemuri R. An Efficient Algorithm for Finding Empty Space for Online FPGA Placement[C]// Proceedings of the 41st Design Automation Conference. NY，USA：[s. n.]，2004

[9] Cui Jin，Deng Qing-xu，He Xiu-qiang，et al. An efficiental agorithm for online management of 2D area of partially reconfigurable FPGAs[C]// 2007 Design，Automati on and Test in Europe Con ference and Exposition. Nice：DAA EDAC IEEE Computer Society TTTC，2007：129-134

[10] Ahmadinia A，Bobda C，Urgen T J. A New Approach for On-line Placement on Reconfigurable Devices[C]// Proceedings of the International Parallel and Distributed Processing Symposium，Reconfigurable Architectures Workshop. Santa FNM，USA：IEEE-CS Press，2004-04

[11] 周学功，梁梁，黄勋章，等. 可重构系统中的实时任务在线调度与放置算法[J]. 计算机学报，2011，48(4)：1901-1909