针对带约束匹配搜索的扩展 Kuhn-Munkres 算法*

王方洋　刘玉铭†
(北京师范大学数学科学学院, 100875, 北京)

摘要　提出了扩展的 Kuhn-Munkres 算法, 可解决带下界约束的局部匹配存在性问题, 即在匹配全集的给定子集中, 搜索得到一个二分图匹配满足其边权和大于给定阈值. 扩展 Kuhn-Munkres 算法构造了一棵以 Kuhn-Munkres 算法中间过程为节点的搜索树, 利用搜索优先级和剪枝, 将算法时间复杂度降低至二分图匹配全集与给定子集差集规模的多项式函数.

关键词　二分图; 最优匹配; Kuhn-Munkres 算法
中图分类号　O29; TP301.6　　**DOI**: 10.12202/j.0476-0301.2019242

0　引言

二分图也被称为“二部图”, 是一种图论模型[1−3], 常用于构建“多对多”的任务分配模型[4−5]、视频检测[7−11]中相邻帧多目标之间关系的数学模型等. 二分图匹配用于描述二分图中不同集合点之间的对应关系.

在任务分配问题中, 给定二分图中不同集合点之间的边权, 同时给定每个任务对每位职员的匹配程度. 二分图匹配是将多任务分给对应职员的分配方案, 最优匹配是所有分配方案中匹配度之和最高的分配方案[4]. 在多目标追踪问题中, 给定相邻帧多目标之间的相似性度量, 则最优匹配是所有目标匹配结果中相似性之和最佳的匹配情况. Kuhn-Munkres 算法[5]或网络流算法[6]能在多项时间内求解二分图最优匹配, 从而在任务分配、多目标追踪等问题中应用较广.

在某些场景下, 由于外部原因导致最优匹配在实际问题中不可行, Kuhn-Munkres 算法无法得到其他匹配结果. 例如: 任务分配问题中, 由于任务或职员自身原因, 致使最优分配方案无法实施; 多目标追踪问题中若使用结构化支持向量机模型[12], 该模型训练时需要求解最优匹配作为新的支持样本, 但最优匹配已存在于旧的支持样本集合中. 以上情况需要求解带一定约束的最优匹配, 此时 Kuhn-Munkres 算法和网络流算法均不适用.

在全体匹配构成集合的一个给定子集上搜索, 得到局部最优匹配是 NP 难题[13]. 若通过约束匹配权值的下界代替寻找最优匹配, 通过算法设计, 在确保较低时间复杂度的情况下, 尽可能搜索权值较大的匹配结果, 即将问题转化为带下界约束的局部较优匹配搜索问题.

本文提出扩展 Kuhn-Munkres 算法(简称 Extended KM 算法), 以求解带下界约束局部较优匹配的搜索问题, 即在给定匹配全集的一个子集中搜索, 得到一个二分图匹配, 满足其边权和大于给定阈值.

1　知识准备

定义 1[1]　称(X,Y,E)为带权二分图, 简记为二分图, 其中 X、Y为 2 个点集, 边集 $E\subseteq\tilde{E}(X,Y)\triangleq\{(x,y,\omega_{x,y}):x\in X,y\in Y\}$, $\omega_{x,y}\in\mathbb{R}$是给定的点$x$到点$y$的边权. 若$|X|=|Y|$, 则称$(X,Y,E)$为平衡二分图.

定义 2　称$\left(X,Y,\tilde{E}\right)$是完全二分图, 若二分图$\left(X,Y,\tilde{E}\right)$满足不同点集之间的任意 2 点都有一条边: $\forall x\in X,y\in Y,\exists!\,(x,y,\omega_{x,y})\in\tilde{E}(X,Y)$.

定义 3[4]　称$\pi_{\widehat{X}}$为二分图(X,Y,E)的一个匹配, 若$\pi_{\widehat{X}}$为定义在$\widehat{X}\subset X$上的单射$\pi_{\widehat{X}}:\widehat{X}\to Y,x\mapsto y$, 满足$\exists\omega_{x,y}\in\mathbb{R},(x,y,\omega_{x,y})\in E$. 匹配全体记做$\mathbb{M}$.

定义 4　称π_X为二分图(X,Y,E)的最大匹配, 简记为π. 最大匹配全体记做$\tilde{M}$.

定义 5　称$\Omega(\pi_{\widehat{X}})\triangleq\sum_{x\in\widehat{X}}\omega_{x,\pi_{\widehat{X}}(x)}$为匹配$\pi_{\widehat{X}}$的权值.

定义 6[5−6]　在$\tilde{\mathbb{M}}$中具有最大权值的匹配为最优

* 国家自然科学基金资助项目(11971065, 11571001); 数字福建智能制造大数据研究所开放课题资助项目(BD201810)
† 通信作者: 刘玉铭(1968—), 男, 博士, 高级工程师. 研究方向: 模糊数学, 图像处理, 智能控制. E-mail: liuyuming@bnu.edu.cn
收稿日期: 2019-09-10

匹配$\pi_y \triangleq \underset{\pi\in\tilde{M}}{\operatorname{argmax}}\, \Omega(\pi)$.

Kuhn-Munkres (Hungarian)算法[5]用于在平衡完全二分图$\left(X,Y,\tilde{E}\right)$中求得最优匹配$\pi_y$, 时间复杂度为$O(n^3)$, 其中$n \triangleq |X|$.

定义 7 给定子集$M \subseteq \tilde{M}$, 称$\pi_{M,\mathrm{o}} \triangleq \underset{\pi\in M\subseteq\tilde{M}}{\operatorname{argmax}}\, \Omega(\pi)$为二分图$(X,Y,E)$的局部最优匹配.

局部最优匹配的求解问题, 实际上在求解第k优匹配, 是NP难题[13].

定义 8 给定下界约束$\widehat{\Omega}\in\mathbb{R}$和子集$M \subseteq \tilde{M}$, 设$\tilde{M}_{\widehat{\Omega}} \triangleq \left\{\pi \in \tilde{M} : \Omega(\pi) > \widehat{\Omega}\right\} \subseteq \tilde{M}$, 匹配$\pi_{(\widehat{\Omega},M)} \in \tilde{M}_{\widehat{\Omega}} \cap M$被称为带下界约束的局部匹配.

1.1 Kuhn-Munkres 算法 记平衡完全二分图(X,Y,E)的点集$X \triangleq \{x_1, x_2, \cdots, x_n\}$, $Y \triangleq \{y_1, y_2, \cdots, y_n\}$, 下标集$\mathbf{N} \triangleq = \{1, \cdots, n\}$.

Kuhn-Munkres算法(X,Y,E)中求得最优匹配$\pi_{\mathrm{o}} \triangleq \underset{\pi\in\tilde{M}}{\arg\max}\, \Omega(\pi)$.

1.1.1 *维护数据* 在算法过程中维护如下内容, $\forall i \in \mathbf{N}$: x_i匹配情况: $m(x_i) \in \mathbf{N} \cup \{\varnothing\}$, 其中$\varnothing$表示$x_i$未匹配到对应的$Y$中的点; x_i和y_i的顶标值: $t(x_i)$、$t(y_i) \in \mathbb{R}$; y_i松弛值: $s(y_i) \in \mathbb{R}$; x_i和y_i的标记: $v(x_i)$、$v(y_i) \in \{T, F\}$, T表示有标记, F表示无标记.

令

差值: $d(x_i, y_j) \triangleq t(x_i) + t(y_j) - \omega_{x_i,y_j}, \forall i, j \in \mathbf{N}$;

导出匹配: 对于匹配情况$m(\cdot)$, 若$m(x_i) \neq \varnothing, \forall i \in \mathbf{N}$, 则称其是完整的, 可导出匹配$\pi_m : x_i \to m(x_i), \forall i \in \mathbf{N}$, 否则称其不完整.

交错路: $Z \triangleq \{z_k\}_{k=1}^{p}$是1个点序列, 满足$z_i \neq z_j$, $\forall i \neq j, i, j \in \mathbf{N}$, $\{z_{2k-1}\}_{k=1}^{\lceil (p+1)/2 \rceil} \subset X, \{z_{2k}\}_{k=1}^{\lfloor p/2 \rfloor} \subset Y$, $m(z_{2k+1}) = z_{2k}$, $\forall k = 1, \cdots, \lfloor (p-1)/2 \rfloor$, 记序列长度$p$为$|\mathbf{Z}|$, 则$\mathbf{Z}$的最后一项为$z_{|\mathbf{Z}|}$.

可增广路: Z是交错路且满足: $|Z|$是偶数、$m(x_i) \neq z_{|Z|}, \forall i \in \mathbf{N}$.

异或匹配情况与可增广路: $m(\cdot) \otimes Z$得到新的匹配情况$m(\cdot)$满足 $m'(x_i) = m(x_i), \forall x_i \notin Z$且$m'(z_{2k-1}) = z_{2k}$, $\forall k = 1, \cdots, |\mathbf{Z}|/2$.

1.1.2 *算法步骤* Kuhn-Munkres算法[5]通过不断寻找从未匹配点$x_i \in X$出发的可增广路, 以扩充当前的匹配情况. 利用深度优先搜索可增广路, 其时间复杂度为$O(n^2)$, 故Kuhn-Munkres算法的总体时间复杂度为$O(n^3)$.

若所有交错路都不是可增广路, 则利用所有未标记Y中的点所记录的最小松弛来修改顶标值和更新松弛值. 即设$D \triangleq \min_{b\in N} v(y_b) = F^{s(y_b)}$, 更新顶标值:

$$\begin{aligned} [t(x_i) - D] &\to t(x_i) \quad \forall v(x_i) = T, \\ [t(y_i) + D] &\to t(y_i) \quad \forall v(y_i) = T. \end{aligned}$$

若$\mathrm{vis}(y_i) = \mathrm{false}$, 则更新松弛值:

$$\begin{aligned} [s(y_i) - D] &\to s(y_i) \quad \forall s(y_i) \neq D, \\ +\infty &\to s(y_i) \quad \forall s(y_i) = D. \end{aligned}$$

Kuhn-Munkres算法在进行过程中保证所有匹配的点x_i、$m(x_i)$都满足顶标和等于边权值, 即差值$\mathrm{d}(x_i, m(x_i)) = t(x_i) + t(m(x_i)) - \omega_{x_i,m(x_i)} = 0$, 从而对于任一完整的匹配情况$m(\cdot)$的导出匹配$\pi_m$满足$\Omega(\pi_m) = \sum_{i=1}^{n} [t(x_i) + t(\pi_m(x_i))]$. 意味着任何时刻算法顶标和都是当前情况下所能找到的匹配权值的上界.

可增广路的性质保证了在修改顶标值时, 被修改的$t(x_i)$比$t(y_i)$数量多1, 从而每次进行修改时, 所有点的顶标和减少当时的修改值D. 在修改顶标值时, 本质上是在降低可能找到的匹配权值的上界, 即不存在一个匹配的权值大于等于修改前的顶标和.

初始化匹配情况$m(x_i) = \mathrm{null}$、顶标值$t(x_i) = \max_{j\in\mathbf{N}} \omega_{i,j}$、$t(y_i) = 0$、松弛值$s(y_i) = +\infty$、标记$v(x_i) = v(y_i) = F$, $\forall i \in \mathbf{N}$, 此时的$\sum_{i=1}^{n} [t(x_i) + t(\pi_m(x_i))]$显然是任意匹配权值的上界.

Kuhn-Munkres算法主要包含2类操作: 1)寻找可增广路; 2)修改顶标. 本质上, Kuhn-Munkres算法在不断进行操作1)当无法找到可增广路时, 进行1次操作2)后继续尝试操作1), 类似于在1棵树上进行深度优先搜索.

2 扩展 Kuhn-Munkres 算法

基于Kuhn-Munkres算法在$\tilde{\mathbb{M}}$中搜索π_{o}的本质, 搜索$\pi_{(\widehat{\Omega},M)}$需要将搜索范围局限在$\tilde{M}_{\widehat{\Omega}} \cap M$中, 因而考虑: 1)必须按1.1.2中的操作1)所可能得到的所有可增广路, 并排除$\tilde{M} - M$的匹配结果; 2)按操作2)修改后顶标仍应大于下界约束$\widehat{\Omega}$.

2.1 维护数据 为方便描述, 树上节点用花体表示, 如$\mathcal{A}$、$\mathcal{B}$、$\mathcal{C}$等.

定义搜索树上任一节点$\mathcal{A}$都包括匹配情况$m_{\mathcal{A}}(x_i)$, 顶标值$t_{\mathcal{A}}(x_i)$、$t_{\mathcal{A}}(y_i)$, 松弛值$s_{\mathcal{A}}(y_i)$, 标记$v_{\mathcal{A}}(x_i)$、$v_{\mathcal{A}}(y_i)$.

若$m_{\mathcal{A}}(\cdot)$是完整的, 其导出的匹配简记为$\pi_{\mathcal{A}} \triangleq \pi_{m_{\mathcal{A}}}$; 反之设$x_{p(\mathcal{A})}$表示节点$\mathcal{A}$在$X$中的最前未匹配点, 其下标$p(\mathcal{A}) \triangleq \min_{a\in N} \{i : m_{\mathcal{A}}(x_i) = \varnothing\}$.

设节点$\mathcal{A}$记录的顶标和$S(\mathcal{A}) \triangleq \sum_{i=1}^{n} [t_{\mathcal{A}}(x_i) + t_{\mathcal{A}}(y_i)]$.

设节点$\mathcal{A}$进行顶标修改所使用的修改值为$D(\mathcal{A}) \triangleq \min\limits_{b\in\mathbf{N}} v_{\mathcal{A}}(y_b) = F^{s_{\mathcal{A}}(y_b)} \in \mathbb{R}\cup\{+\infty\}$.

2.2 根节点及初始化 根节点记为$\mathcal{R}$, 初始化与 Kuhn-Munkres 算法一致: $m_{\mathcal{R}}(x_i)=\varnothing$, $t_{\mathcal{R}}(x_i)=\max\limits_{j\in N}\omega_{i,j}$, $t_{\mathcal{R}}(y_i)=0$, $v_{\mathcal{R}}(x_i)=v_{\mathcal{R}}(y_i)=F$, $s_{\mathcal{R}}(y_i)=+\infty$, $\forall i\in\mathbf{N}$.

2.3 节点扩展和算法结束 对$\mathrm{m}_{\mathcal{A}}(\cdot)$不完整的节点$\mathcal{A}$, 定义 2 种扩展方式以获得子节点$\mathcal{B}$.

左扩展$L(\mathcal{A})$: 从节点$\mathcal{A}$寻找可增广路, 将搜索过程中记录的标记、松弛更新至节点$\mathcal{A}$. 若$x_{p(\mathcal{A})}$存在可增广路$\mathbf{Z}$, 子节点$\mathcal{B}$保留$\mathcal{A}$的顶标、标记、松弛等, 更新匹配情况$m_{\mathcal{A}}(\cdot)\otimes\mathbf{Z}\to m_{\mathcal{B}}(\cdot)$; 若$x_{p(\mathcal{A})}$不存在可增广路, 则称$\mathcal{A}$不可进行左扩展.

右扩展$R(\mathcal{A})$: 对于已经尝试过$L(\mathcal{A})$的节点$\mathcal{A}$, 可考虑进行$R(\mathcal{A})$: 若$D_{\mathcal{A}}<S(\mathcal{A})-\widehat{\Omega}<+\infty$, 子节点$\mathcal{B}$保存对$\mathcal{A}$修改顶标后的顶标和松弛, 保留$\mathcal{A}$的匹配情况, 重置标记$v_{\mathcal{B}}(x_i)=v_{\mathcal{B}}(y_i)=F, \forall i\in\mathbf{N}$; 若$S(\mathcal{A})-D_{\mathcal{A}}\leqslant\widehat{\Omega}$, 则称$\mathcal{A}$不必进行右扩展(特别地, 当$D_{\mathcal{A}}=+\infty$时称$\mathcal{A}$不可进行右扩展).

$L(\mathcal{A})$在搜索可增广路时可能存在多条可增广路, 每条可增广路对应 1 个子节点, 即$L(\mathcal{A})$的结果为 1 个包含若干个子节点的集合.

与 Kuhn-Munkres 算法一致, 对于任一节点$\mathcal{A}$, 顶标和$S(\mathcal{A})$是以$\mathcal{A}$为根节点的子树上所能找到的所有匹配的权值的上界. $L(\mathcal{A})$操作不修改顶标值, 只用于在匹配权值等于$S(\mathcal{A})$的情况下寻找可增广路, 以探索进一步匹配的可能; $R(\mathcal{A})$操作将会在无法进一步匹配的情况下降低对匹配权值的上界, 即用$D(\mathcal{A})$修改顶标和$S(\mathcal{A})$, 以给予$L(\mathcal{A})$更多选择, 即

$$\begin{cases} S(L(\mathcal{A}))=S(\mathcal{A}), \\ S(R(\mathcal{A}))=S(\mathcal{A})-D(\mathcal{A}). \end{cases}$$

当节点$\mathcal{A}$已具有完整匹配$\pi_{\mathcal{A}}$, 且$\pi_{\mathcal{A}}\in M$, 易知$\Omega(\pi_{\mathcal{A}})>\widehat{\Omega}$, 则$\pi_{\mathcal{A}}$即为所求匹配.

2.4 可增广路搜索 Kuhn-Munkres 算法只需要搜索得到 1 条可增广路便可继续, 与之不同的是, $L(\mathcal{A})$要求搜索得到所有的可增广路.

$L(\mathcal{A})$要求$m_{\mathcal{A}}(\cdot)$不完整, 即存在$p(\mathcal{A})\in\mathbf{N}$, $L(\mathcal{A})$将搜索从未匹配点$x_{p(\mathcal{A})}\in X$出发的可增广路, 具体步骤如下:

Step 1 清除所有标记, 置$v(x_i)=v(y_i)=F, \forall i\in\mathbf{N}$.

Step 2 为寻找所有可增广路$\tilde{Z}\triangleq\{\mathbf{Z}_1,\mathbf{Z}_2,\cdots\}$, 设当前搜索中的交错路为$\mathbf{Z}$, 一旦$Z$被更新为可增广路就将其加入$\tilde{Z}$, 初始时$\tilde{Z}=\varnothing, \mathbf{Z}=\{z_1\}, z_1=x_{p(\mathcal{A})}$, 重复 Step 2.1、2.2.

Step 2.1 标记$z_{|\mathbf{Z}|}$, 置$\mathrm{vis}(z_{|\mathbf{Z}|})=T$, 有$z_{|\mathbf{Z}|}\in X$.

Step 2.2 遍历所有未标记的Y中, 且差值$d(z_{|\mathbf{Z}|},y_b)$非负的点$\{y_b: d(z_{|\mathbf{Z}|},y_b)\geqslant 0, v_{\mathcal{A}}(y_b)=F, b\in\mathbf{N}\}$, 重复 Step 2.2.1、2.2.3, 当前遍历的点记为y_b.

Step 2.2.1 若$d(z_{|\mathbf{Z}|},y_b)>0$, 更新松弛值$\min\{d(z_{|\mathbf{Z}|},y_b)\, s_{\mathcal{A}}(y_b)\}\to s_{\mathcal{A}}(y_b)$.

Step 2.2.2 若$d(z_{|\mathbf{Z}|},y_b)=0$, 且$\exists x_c$, 使得$m_{\mathcal{A}}(x_c)=b$, 即$y_b$已被$x_c$匹配, 则标记$y_b$, 即设置$v_{\mathcal{A}}(y_b)=T$, 并将$y_b, x_c$加入$\mathbf{Z}$, 更新$\mathbf{Z}\cup\{y_b,x_c\}\to\mathbf{Z}$, 此时$z_{|\mathbf{Z}|}$变为$x_c$, 回到 Step 2.1.

Step 2.2.3 若$d(z_{|\mathbf{Z}|},y_b)=0$, 且$\nexists x_c$使得$m_{\mathcal{A}}(x_c)=b$, 即$y_b$尚未被匹配, 将$y_b$加入$\mathbf{Z}$, 更新$\mathbf{Z}\cup\{y_b\}\to\mathbf{Z}$, 则$\mathbf{Z}$为可增广路, 进行检查: 如果$m_{\mathcal{A}}\otimes\mathbf{Z}$是完整的、且导出匹配$\pi_{m_{\mathcal{A}}\otimes Z}\notin M$, 则执行 Step 2.2.3.1, 否则执行 Step 2.2.3.2.

Step 2.2.3.1 可增广路$\mathbf{Z}$不符合要求, 回到 Step 2.2 继续遍历下一个y_b.

Step 2.2.3.2 可增广路$\mathbf{Z}$符合要求, 标记y_b即设置$v(y_b)=T$, 并将可增广路$\mathbf{Z}$加入$\tilde{Z}$, 即更新$\tilde{Z}\cup\{\mathbf{Z}\}\to\tilde{Z}$.

Step 2.3 若 Step 2.2 中未进行过下述 3 类更新: 更新松弛(Step 2.2.1), 找到新的可增广路以更新$\tilde{Z}$(Step 2.2.3.1), 在更长的可增广路搜索中更新松弛或更新$\tilde{Z}$(Step 2.2.2 中进行 Step 2.2.1 或 Step 2.2.3.2), 则说明当前的可增广路Z是无可为继的(意味着即使修改顶标, 任何包含当前Z的可增广路都因不符合要求而不能被合并至匹配情况$m_{\mathcal{A}}$), 考虑如下 2 种情况:

Step 2.3.1 如果目前$|\mathbf{Z}|=1$, 即$\mathbf{Z}=\{x_{p(\mathcal{A})}\}$, 则节点$\mathcal{A}$不存在可增广路;

Step 2.3.2 否则$z_{|\mathbf{Z}|-1}\in Y$, $z_{|\mathbf{Z}|}\in X$, 进行回滚, 重置标记$v_{\mathcal{A}}(z_{|\mathbf{Z}|-1})=v_{\mathcal{A}}(z_{|\mathbf{Z}|})=F$.

2.5 搜索优先级及优化剪枝 考虑如下可优化情况:

针对$L(\mathcal{A})$: 应基于父节点$\mathcal{A}$的松弛进行L, 而不是重置松弛后重新维护新的松弛值.

针对R: 一方面, 无需连续地进行R; 另一方面, 记$f(\mathcal{A})$是以节点$\mathcal{A}$为根的子树, 则$S(\mathcal{A})\geqslant\max\limits_{\mathcal{B}\in f(\mathcal{A})}\Omega(\pi_{\mathcal{B}})$, 于是子树$f(\mathcal{A})$中所可能得到的匹配的权值不超过$S(\mathcal{A})$, 从而$S(\mathcal{A})-D_{\mathcal{A}}<\widehat{\Omega}$的节点$\mathcal{A}$不必进行$R$.

与 Kuhn-Munkres 算法类似, 应当优先进行$L(\mathcal{A})$, 其次考虑到$S(\cdot)$表明了节点及其所有子节点所能得到匹配的权值的上界, 则应当优先对$S(\mathcal{A})-D(\mathcal{A})$中较大者进行$R(\mathcal{A})$. 具体地, 对当前搜索树重复如下 2 步操作:

1)深度优先搜索: 进行$p(\mathcal{A})$优先的左扩展操作,

即按$p(\mathcal{A})$从大到小的顺序进行尽可能多次$L(\mathcal{A})$.

2)一次松弛:选择$p(\mathcal{A})$最大的、有必要右扩展的节点$\mathcal{A}$进行$R(\mathcal{A})$(有必要是指不考虑不必进行右扩展的节点).

$p(\mathcal{A})$意味着当前搜索节点$\mathcal{A}$距离得到一个完整匹配尚需的L次数,以$p(\mathcal{A})$大者优先搜索,可以尽快搜索得到合适的解,而不将时间浪费在搜索局部最优匹配这一结果上.

3 时间复杂度分析

对于平衡完全二分图$\left(X,Y,\tilde{E}\right)$,记$n\triangleq|X|$,$m\triangleq|M|$,$p\triangleq\left|\tilde{M}_{\widehat{\Omega}}\right|$,则$\left|\tilde{M}_{\widehat{\Omega}}-M\right|\leqslant p-m$.记$O(c)$为检查一个匹配是否在$M$中的时间复杂度.

下面从2方面说明Extended KM算法的时间复杂度.

3.1 一般情形

命题1 $R(\mathcal{A})$操作的时间复杂度为$O(n)$.

证明 $R(\mathcal{A})$修改所有点的顶标值和松弛值,复杂度为$O(n)$.

命题2 若$p(\mathcal{A})=n$,$\mathcal{A}$的子树$f(\mathcal{A})$所包含的由L产生的叶子节点$E(\mathcal{A})=\{L(\mathcal{A})\}\cup\{L(R(\mathcal{A}))\}$,且其集合大小有上界,即$|E(\mathcal{A})|=O(n)$.

证明 由定义显然有$E(\mathcal{A})\subseteq\{L(\mathcal{A})\}\cup\{L(R(\mathcal{A}))\}$.由$p(\mathcal{A})=n$可知,对于节点$\mathcal{A}$至多进行1次$L$且不可连续进行$R$,即$E(\mathcal{A})\supseteq\{L(\mathcal{A})\}\cup\{L(R(\mathcal{A}))\}$.进而$F$由$|\{R(\mathcal{A})\}|\leqslant 1$及$|\{L((\mathcal{B}))\}|\leqslant n,\forall\mathcal{B}$知$|\{L(R(\mathcal{A}))\}|=O(1\times n)$,从而$|E|\leqslant|\{L(\mathcal{A})\}|+|\{L(R(\mathcal{A}))\}|=O(2n)=O(n)$.

命题3 当$p(\mathcal{A})<n$时,$L(\mathcal{A})$操作的时间复杂度为$O(n^2)$.

证明 在一次L操作中,标记$v_{\mathcal{A}}(y_i),\forall i\in\mathbf{N}$初始化为$F$,而后将仅枚举$\{y_b:v(y_b)=F,b\in\mathbf{N}\}$,并将枚举点$y_b$标记$v_{\mathcal{A}}(y_b)=T$.$p(\mathcal{A})<n$时不会发生回滚操作,标记$v_{\mathcal{A}}(y_i),\forall i\in\mathbf{N}$只发生一次变化,时间复杂度为$O(n^2)$.

命题4 $L(\mathcal{A})$操作的时间复杂度为$O(n^2+n^2\times(p-m)\times(n+c))$.

证明 仅当$p(\mathcal{A})=n$时,$L(\mathcal{A})$操作可能产生至多$\left|\tilde{M}_{\widehat{\Omega}}-M\right|$次回滚:将$v_{\mathcal{A}}(y_b)$重置为false,并且需要检查匹配是否在$M$中.回滚操作的时间复杂度为$O((p-m)\times(n+c))$.考虑到命题3,总时间复杂度为$O(n^2+n^2\times(p-m)\times(n+c))$.

命题5 从根节点$\mathcal{R}$搜索至节点$\mathcal{A}$使得$p(\mathcal{A})=n$,时间复杂度为$O(n^3)$.

证明 显然$p(\mathcal{R})=1$,节点$\mathcal{A}=(L^{a_i}\circ R)^b(\mathcal{R})$,其中$\circ$表示复合操作符,$L^{a_i}$表示$a_i$次$L$操作复合,且$i=1,\cdots,b$.若$p(\mathcal{A})=n$,则$\sum_{i=1}^{b}a_i=n-1$,式中所有$n-1$次$L$操作都不发生回滚.由命题3可知,搜索得到节点$\mathcal{A}$的时间复杂度为$O\left(n^2\times\sum_{i=1}^{b}a_i\right)=O(n^3)$.

综上,搜索$\pi_{(\widehat{\Omega},M)}$的算法总时间复杂度为$O(n^3+n^2+n^2\times(p-m)\times(n+c))=O(n^2\times\max\{n,c\}\times\max\{p-m,1\})$.

3.2 退化情形

若令$\widehat{\Omega}=-\infty M=\tilde{M}_{\widehat{\Omega}}$,则$\tilde{M}_{-\infty}=\tilde{M}$.由Extended KM搜索算法流程可知,此时搜索得到$\pi_{(-\infty,\tilde{M})}$就是最优匹配$\pi_o$.

命题6 Extended KM算法求解$\pi_{(-\infty,\tilde{M})}$的时间复杂度为$O(n^3)$.

证明 由命题5,得到节点$\mathcal{A}$使得$p(\mathcal{A})=n$,时间复杂度为$O(n^3)$.自节点$\mathcal{A}$进行$L(\mathcal{A})$操作,由于$M=\tilde{M}_{\widehat{\Omega}}$,故而不会发生回滚操作,即得到$\pi_{(-\infty,\tilde{\mathbb{M}})}$.由命题4可知,此$L$操作时间复杂度为$O(n^2+n^2\times0\times(n+c))=O(n^2)$.综上可得$\pi_{(-\infty,\tilde{M})}$的时间复杂度为$O(n^3)$.

事实上,算法将在第一次搜索得到具有完整匹配$m_{\mathcal{A}}(\cdot)$的节点$\mathcal{A}$时结束,与Kuhn-Munkres算法流程完全一致,易知其时间复杂度为$O(n^3)$.

4 对比试验

给定的平衡完全二分图$\left(X,Y,\tilde{E}\right)$,给定下界约束$\widehat{\Omega}\in\mathbb{R}$和子集$M\subseteq\tilde{\mathbb{M}}$,求解$\pi_{(\widehat{\Omega},M)}\in\tilde{\mathbb{M}}_{\widehat{\Omega}}\cap M$.

一方面,使用Extended KM算法解$\pi_{(\widehat{\Omega},M)}$作为实验组;另一方面,由于目前没有其他多项式算法求解$\pi_{(\widehat{\Omega},M)}$,本章使用随机搜索算法作为对照组,即不断生成新的随机匹配,并检查其是否可作为答案.

下面设计对比试验,以对比2种算法的耗时、得到解的匹配权值:前者表征了算法时间复杂度,越低则性能越佳;后者表征了算法贴近局部最优匹配的程度,权值$\Omega\left(\pi_{(\widehat{\Omega},M)}\right)$越大,则结果效果越佳.

4.1 实验设计

考虑到随机搜索算法具有随机性,为了更好地反映实验组在上述2类评价指标的对比结果,先使用实验组算法连续生成若干匹配结果作为集合$\tilde{M}-M$,并求得下1个匹配结果$\pi_{(-\infty,M)}\in\tilde{M}_{-\infty}\cap M$,再设下界约束为$\widehat{\Omega}\triangleq\Omega(\pi_{(-\infty,M)})$,最后使用$M$和$\widehat{\Omega}$对实验组和对照组进行测试.其中"连续生成"是指对于给定的$K\in\mathbb{N}$,使用如下方式生成K个匹配结果$\{\pi^{(k)}\}_{k=1,\cdots,K}$:利用实验组算法求解$\pi^{(k)}\triangleq\pi_{(-\infty,M_k)}\in\tilde{M}_{-\infty}\cap M_k$,其中$M_0=\tilde{M},M_k=M_{k-1}-\{\pi^{(k-1)}\},\forall k=1,\cdots,K$,即不断将所得匹配从考虑的匹配集合中删去,再求符合要求的匹配.

4.2 实验结果

4.2.1 实验1 分别对$n=4,5,\cdots,10$随机生成1 000个

二分图，分别取$K=1,10,20$，对照组使用随机搜索算法（random search）、实验组使用 Extended KM 算法分别进行上述实验，绘制算法平均运行时长关于n的折线图，如图 1 所示．图 1 中已标出$K=20$时的算法耗时均值．

由图 1 可以看出：随n增大时，随机搜索算法的时间开销按指数级增长，而 Extended KM 算法时间开销变化不大．

4.2.2　实验 2　进一步地，分别取$n=11,12,\cdots,100$，$K=1,5,15,20$，此时对照组随机搜索算法时间开销过高，难以进行实验，但实验组使用 Extended KM 算法可得到结果．绘制实验组平均运行时长关于n的折线图，如图 2 所示，保留了实验 1 中$n\leqslant 10$、$K=20$情形下的对照组随机搜索算法的时间开销作为对照．

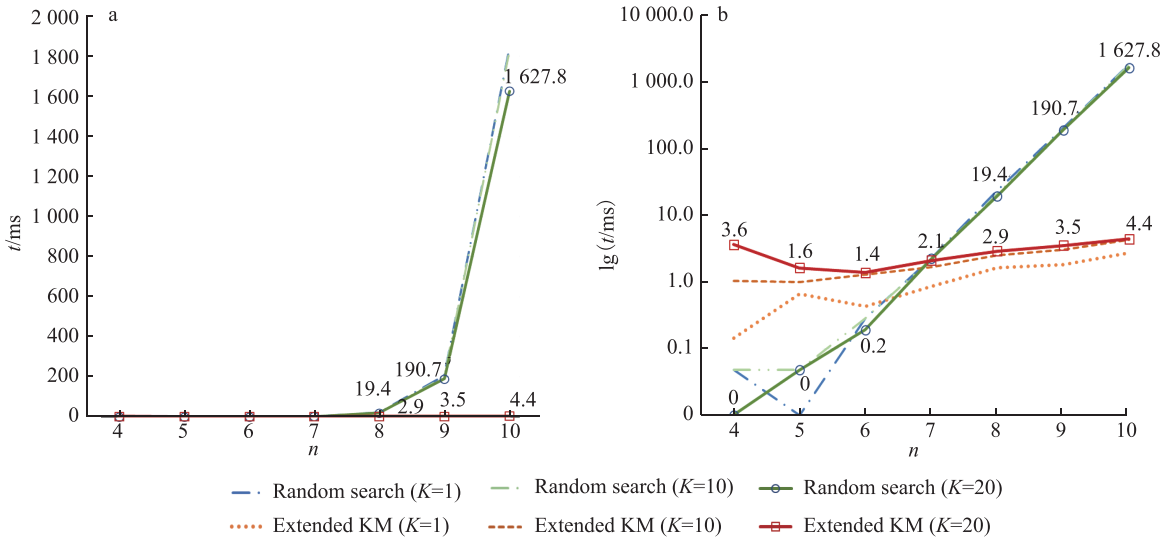

图 1　不同K值下 **Extended KM** 算法与随机搜索算法的运行时间关于$n\in[4,10]$的变化趋势

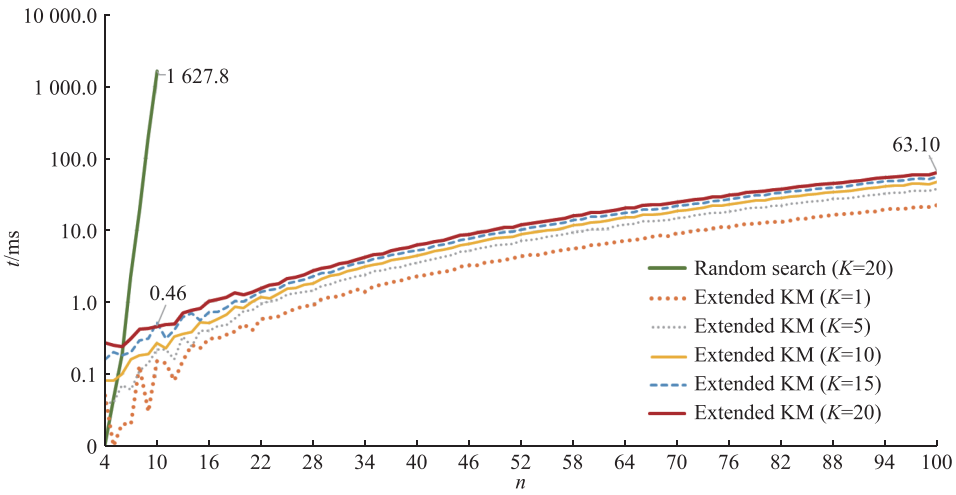

图 2　不同K值下 **Extended KM** 算法运行时间关于$n\in[4,100]$的变化趋势（纵轴为对数坐标轴）

由图 2 可以看出：随n逐渐增大，Extended KM 算法性能较优，时间复杂度远低于随机搜索算法的指数级；并且在K增大时，时间开销变化不大．

综上，实验表明 Extended KM 算法搜索得到$\pi_{(\widehat{\Omega},M)}\in\tilde{M}_{\widehat{\Omega}}\cap M$的效率较高，时间复杂度基本为多项式级别．

5　结论

本研究提出 Extended KM 算法，在平衡完全二分图$\left(X,Y,\tilde{E}\right)$中，给定下界约束$\widehat{\Omega}\in\mathbb{R}$和子集$M\subseteq\tilde{\mathbb{M}}$，寻找一个带下界约束的局部较优匹配，即$\pi_{(\widehat{\Omega},M)}\in\tilde{\mathbb{M}}_{\widehat{\Omega}}\cap M$．

Extended KM 算法有以下特点：

1）Extended KM 算法蕴含最优匹配搜索的 Kuhn-

Munkres算法，Extended KM算法在求解最优匹配$\pi_o=\pi_{(-\infty,\varnothing)}$时与Kuhn-Munkres算法时间复杂度相同.

2)局部最优匹配$\pi_{M,o}$是NP难问题，但Extended KM算法可以在多项式级别的时间复杂度下找到一个局部较优匹配$\pi_{(\hat{\Omega},M)}$，并且满足给定的下界约束.这一特性可以在多种实际问题中发挥重要作用.

3)Extended KM算法虽然不一定能找到最优的解，但搜索过程是在尽快地寻找尽量优的匹配.

事实上，由于Extended KM算法时间复杂度正比于$|\tilde{M}_{\hat{\Omega}}-M|$，时间开销将随着$|\tilde{M}_{\hat{\Omega}}-M|$的增大而增大，且$|M|$较小时，$|\tilde{M}_{\hat{\Omega}}-M|$是关于$n$的指数级规模.若$|M|$不足够小，可以应用其他搜索方法(如随机搜索算法或枚举法等)，结合Extended KM算法，尽可能覆盖到更广范围的情形.

6 参考文献

[1] KAY E. Graph theory with applications[J]. Journal of the Operational Research Society, 1977, 28(1): 237

[2] DIESTEL R. Graph theory[J]. Mathematical Gazette, 2000, 173(502): 67

[3] KORTE B, VYGEN J. Combinatorial optimization: theory and algorithms[M]. Berlin: Springer, 2000

[4] LOVASZ L, PLUMMER M D. Matching theory[M]. Amsterdam: Elsevier, 1986, 29

[5] KUHN H W. The Hungarian method for the assignment problem[J]. Naval Research Logistics, 1955, 2(1/2): 83

[6] MYERS B R. Graphs, networks and algorithms[J]. Proceedings of the IEEE, 1981, 70(7): 781

[7] KIM S, KWAK S, FEYEREISL J, et al. Online multi-target tracking by large margin structured learning[C]//Asian Conference on Computer Vision. Berlin: Springer, 2012: 98

[8] HUANG C, WU B, NEVATIA R. Robust object tracking by hierarchical association of detection responses[C]// Computer Vision-ECCV 2008, 10th European Conference on Computer Vision, Proceedings, Part II, Marseille, France: Springer-Verlag, 2008: 788

[9] LI Y, HUANG C, NEVATIA R. Learning to associate: HybridBoosted multi-target tracker for crowded scene[C]//2009 IEEE conference on computer vision and pattern recognition, Fontainebleau Resort. Miami Beach: IEEE, 2009: 2953

[10] KUO C H, HUANG C, NEVATIA R. Multi-target tracking by on-line learned discriminative appearance models[C]//2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. New Jersey: IEEE, 2010: 685

[11] KUO C H, NEVATIA R. How does person identity recognition help multi-person tracking?[C]//IEEE Conference on Computer Vision & Pattern Recognition. New Jersey: IEEE, 2011: 1217

[12] TSOCHANTARDIS I, THOMAS H, THORSTEN J, et al. Support vector machine learning for interdependent and structured output spaces[C]//Proceedings of the twenty-first international conference on Machine learning. New York: Association for Computing Machinery, 2004: 104

[13] VALIANT L G. The complexity of computing the permanent[J]. Theoretical Computer Science, 1979, 8(2): 189

Extended Kuhn-Munkres algorithm for constrained matching search

WANG Fangyang LIU Yuming†

(School of Mathematical Sciences, Beijing Normal University, 100875, Beijing, China)

Abstract With Kuhn-Munkres algorithm no optimal matching is founded on matching subset. An extended Kuhn-Munkres algorithm is proposed here to solve this problem of local matching with lower bound constraints, to search for a bipartite graph matching with edge-weights greater than a given threshold from a given matching subset. At present, there is no polynomial algorithm to solve this problem while the time complexity of general search algorithm is exponential. Extended Kuhn-Munkres algorithm constructs a search tree with an intermediate process of Kuhn-Munkres algorithm as nodes. With search priority and pruning, time complexity of searching for result matching is reduced to a polynomial function about the size of bipartite graph and the size of the difference set between the whole matching set and a given matching subset.

Keywords bipartite graph; optimal matching; Kuhn-Munkres algorithm

【责任编辑：陆有忠】