湖南大學

体系结构报告

题 目：计算机体系结构综述报告

学 生 姓 名 谭 珍 琼

学 生 学 号 S191000912

专 业 班 级 计算机科学与技术

完 成 日 期 2019年12月27日

基于二分图的学习任务分配

摘要

本文对现有的任务分配方法进行调研，尝试在此基础上将任务分配技术应用到学生学习的场景，帮助学生合理安排和调整各项学习任务、制定合适的时间表，以达到适合学生个性化情况的最佳学习效果。

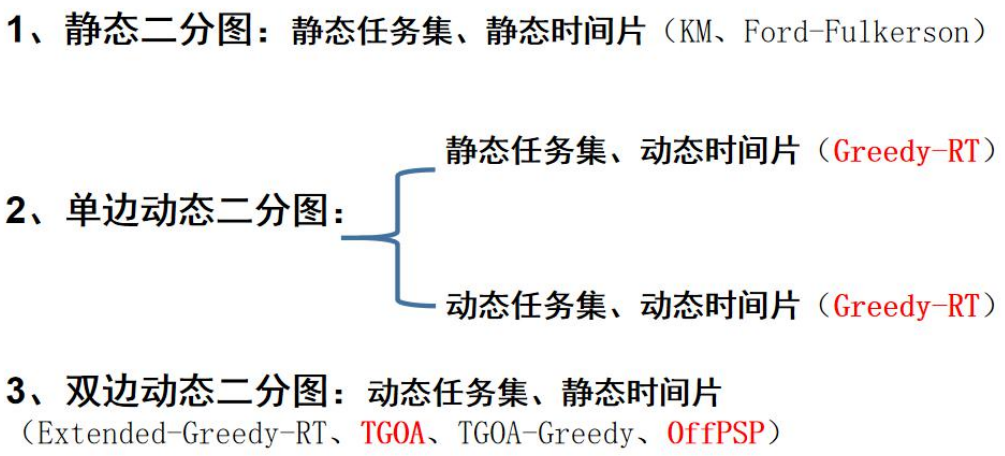
在本篇文章中，我将概述什么是二分图、什么是学习任务分配以及怎样使用二分图的方法解决学习任务分配的问题。根据学生学习的真实场景，本文主要分为三类方法：静态二分图匹配、单边动态二分图匹配和双边动态二分图匹配。

关键字：二分图、学习任务分配、最优匹配、学习效率

1引言

近年来随着学习资源、学习方式的多样化，学生普遍存在不懂得合理安排学习任务的现象，导致最后又急又赶的完成学习任务，既不能达到良好学习效果也没有很好的完成学习任务。本文将任务分配方法应用到学生学习的场景，帮助学生合理安排和调整各项学习任务、制定合适的时间表，以达到适合学生个性化情况的最佳学习效果。给定K名学生，每名学生都有N个空闲的时间片和M个需要完成的任务，每个任务在不同时间片完成有不同的学习效率，如何为每名学生在合适的时间片中安排好所有学习任务，使得整个学习效率最好。

根据现有研究方法，结合学生学习的真实场景，本文主要从静态二分图任务分配、单边动态二分图任务分配及双边动态二分图任务分配三个方面进行了阐述，介绍了任务分配的基本概念、二分图的基本概念、常用匹配算法等。本文分类索引如下图所示：



2 国内外研究情况

目前，关于使用二分图的方法进行任务分配的研究已经非常成熟，根据二分图的顶点是否能变化分为静态二分图、单边动态二分图、双边动态二分图；根据连接顶点的边是否带权重，分为加权二分图和不加权二分图；根据匹配目标分为二分图最大最小匹配、最优匹配、最大匹配、完全匹配等；根据应用到不同的场景又有众包任务分配、在线任务分配、多主体任务分配，其中众包任务分配在当前是一项非常热点的研究。

但是，目前还没有人使用二分图的方法进行学习任务分配，本文是首次提出基于二分图的学习任务分配应用到学生学习的场景。

3 基本概念

3.1 学习任务分配基本概念

任务分配是指给定多个任务集和工人集，目标是根据相应规则将每个任务分配给合适工人。学习任务分配是由任务分配的概念应用到学生学习场景中去所得：给定多个学生学习任务集合和时间片集合，根据每个学生的学习效率，目标是把每个学习任务分配给合适的时间片，使每个学生的学习效果达到最优。

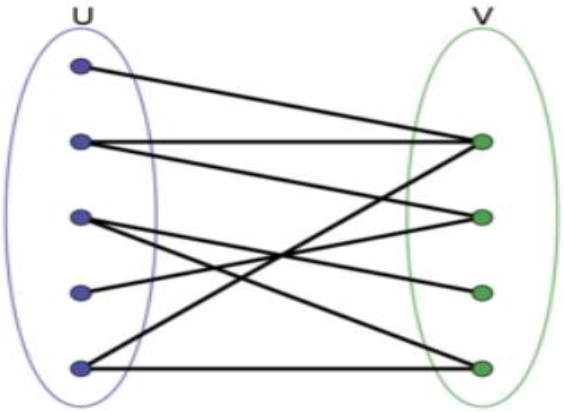
是否合适，取决于下面两个目标：

（1）使学生能最快速的完成多个学习任务（任务最快完成）

（2）使学生能最高效的完成多个学习任务（学习效果收益最好）

3.2 二分图基本概念

二分图又称作二部图，是图论中的一种特殊模型。 设 G=(V,E)是一个无向图，如果顶点 V 可分割为两个互不相交的子集(A,B)，并且图中的每条边（i，j）所关联的两个顶点 i 和 j 分别属于这两个不同的顶点集(i in A,j in B)，则称图 G 为一个二分图，如下图所示：



二分图最大匹配：一个图所有匹配中所含匹配边数最多的匹配，称为这个图的最大匹配。

二分图最优匹配：是指在给定带权二分图上求出一个最大匹配，使得所有匹配边权值之和最大。

根据连接两个顶点的边是否带权重，可将二分图分为加权二分图和不加权二分图；根据二分图中顶点个数是否固定，又可将其分为静态二分图、单边动态二分图、双边动态二分图；本文将研究在静态模式下和动态模式下的加权二分图最大匹配问题。

3.3 任务分配的质量控制问题(权重，学习效率问题)

质量控制与任务分配问题相互影响。质量控制一方面是任务分配问题的重要优化目标，为任务分配提供指导；另一方面，质量控制结果的好坏也依赖于任务分配策略制定得是否合理。因此，两者互相影响且相互依存。

3.4 分配的约束条件

任务必须分配给在其发布日期之后，截止日期之前的时间片

任务不能分配给时长小于它的时间片

已分配的任务（时间片）不再参与分配

3.5 分配效果评价指标

（1）最终任务完成时间：与截止时间相比，越早完成越好

（2）最终花费学习时间：花费时间越少越好

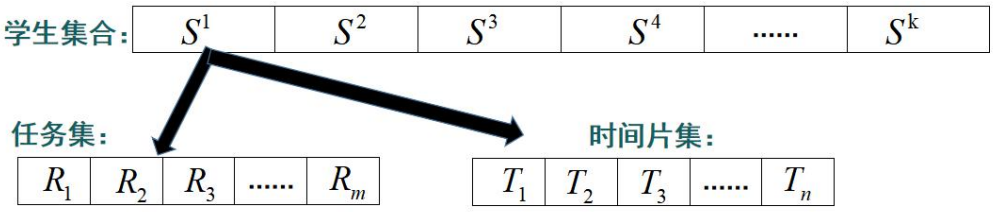
（3）最终学习效率：学习效率越高越好

4 问题定义

学生集合：$S=\{s_1,s_2,.....,s_k\}$

任务集合：$R_i=\{r_i^1,r_i^2,.....,r_i^m\}$

时间片集合：$T_i=\{t_i^1,t_i^2,.....,t_i^n\}$



学生对象具有任务、时间片、效率表三个属性

（1）任务：id 编号、description 任务描述、duration 任务时长、starth 任务发布时间、deadline 任务截止时间、flag 任务是否安排标记

（2）时间片：id 编号 、duration 时间片时长 、starth 开始时间 、endh 结束时间 、flag 任务是否安排标记

（3）基于时间的学习任务效率曲线：（每名学生的每个任务的学习效率曲线都不一样）

（4）任务—时间片效率表：根据任务效率曲线，任务与每个时间片生成不同的学习效率

R_1

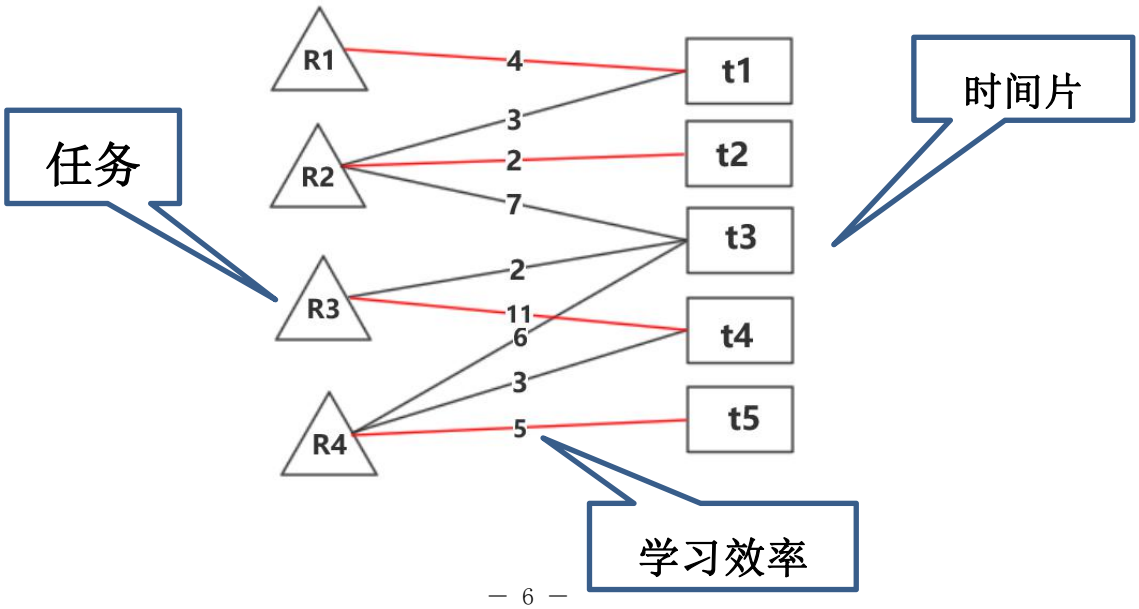
id	release	duration	deadline

T_1

id	duration	starth	ended

根据学生任务分配中所描述的实际问题：给定 K 名学生，每名学生都有 N 个空闲的时间片和 M 个需要完成的任务，每个任务在不同时间片完成有不同的学习效率，如何为每名学生在合适的时间片中安排好所有学习任务，使得整个学习效率最好。

显然，我们只需要构造这样的数学模型，以一名学生 S 为例，学生的任务集作为左顶点集 V，时间片集作为右顶点集 E，将任务与能在其完成的时间片之间连边，完成的学习效率作为权重 W，得二分图 G，然后在 G 中找到一个边的子集，使得每个顶点只能与一条边关联，并且所含匹配边数最多、所得权重之和最大，如下图所示：



5 静态二分图匹配

5.1 静态任务集、静态时间片

给定学生的学习任务/时间片集合的全部信息，针对指定效率，将学习任务一次性分配给合适的时间片，整个分配过程中学生的任务集、时间片集都是固定的，不再发生任何变更。在进行任务分配之前，所有任务和时间片的信息都是已知的，这些信息中没有不确定性。

5.2 最大化权重二分图匹配的定义

根据静态学习任务分配的定义，可以将其建模成经典的最大化权重二分图匹配 (MWBM)问题，将学习任务和时间片视为二分图中左右两个不相交的点集，将每个学习任务可以执行的时间范围内的时间片与该任务在二分图内构建边，该任务在该时间片完成的学习效率构成权重，寻找带权二分图的最佳匹配方法。特点是实现简单，往往可以很容易地得到全局最优目标，但是效果有限。

5.3 匹配算法

5.3.1 KM 算法

KM 算法是一种计算机算法，功能是求完备匹配下的最大权匹配。在一个二分图内，左顶点为 X，右顶点为 Y，现对于每组左右连接 XiYj 有权 wij，求一种匹配使得所有 wij 的和最大。

步骤：（1）初始化可行顶标的值；

（2）用匈牙利算法寻找完备匹配；

（3）若未找到完备匹配则修改可行顶标的值；

（4）重复（2）（3）直到找到相等子图的完备匹配为止。

5.3.2 Ford-Fulkerson 算法

Ford-Fulkerson 算法（FFA）是一种贪婪算法，用于计算流网络中的最大流量。思想是：只要存在从源（起始节点）到接收器（端节点）的路径，在路径中的所有边缘上具有可用容量，我们就沿着其中一个路径发送流。 然后我们找到另一条路径，依此类推。 具有可用容量的路径称为扩充路径。

6 单边动态二分图匹配

6.1 动态任务集、静态时间片

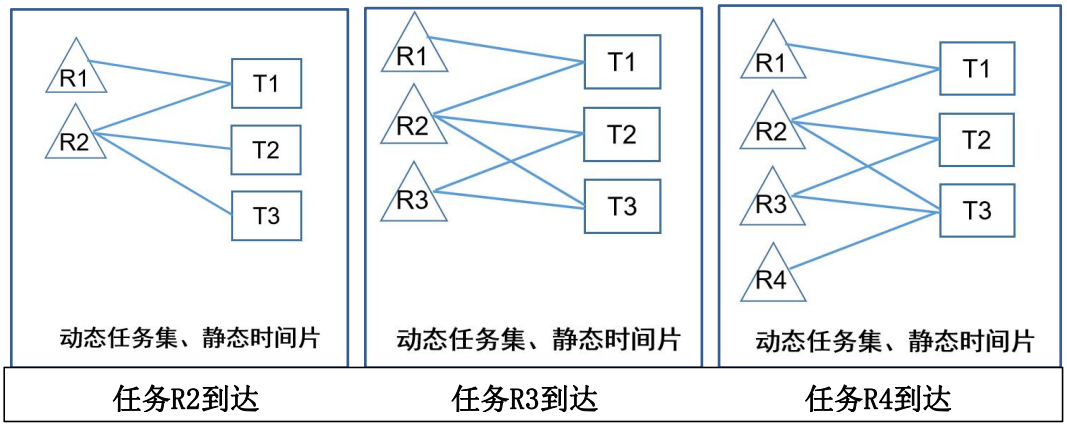
给定学生时间片集的全部信息，学习任务动态出现在任务集中，针对指定效率，如何为新出现的学习任务分配最合适的时间片（已分配的任务—时间片匹配对不再改变）。

6.2 静态任务集、动态时间片

给定学生任务集的全部信息，时间片集随学生状态动态更新，针对指定效率，如何把学习任务分配给新的的时间片集（已分配的任务—时间片匹配对不再改变）。

6.3 单边动态加权二分图匹配的定义

根据 5.1-5.2 学习任务分配的定义，可以将其建模成单边动态加权二分图匹配问题，以静态任务集、动态时间片、静态效率为例构建：二分图 G =（L，R，E，U），其左顶点 L 任务集是事先已知的，但是右顶点 R 时间片集是未知的，时间片 R 动态一一到达，且 r∈R，入射到 r 的边（l，r）∈E 及其对应边的权重 U（l，r）∈U，并且时间片 r 必须匹配左顶点任务 l 或此后保持不匹配。 目标是匹配边的总权重之和最大化。 任务集动态到来过程如下图所示：



6.4 匹配算法

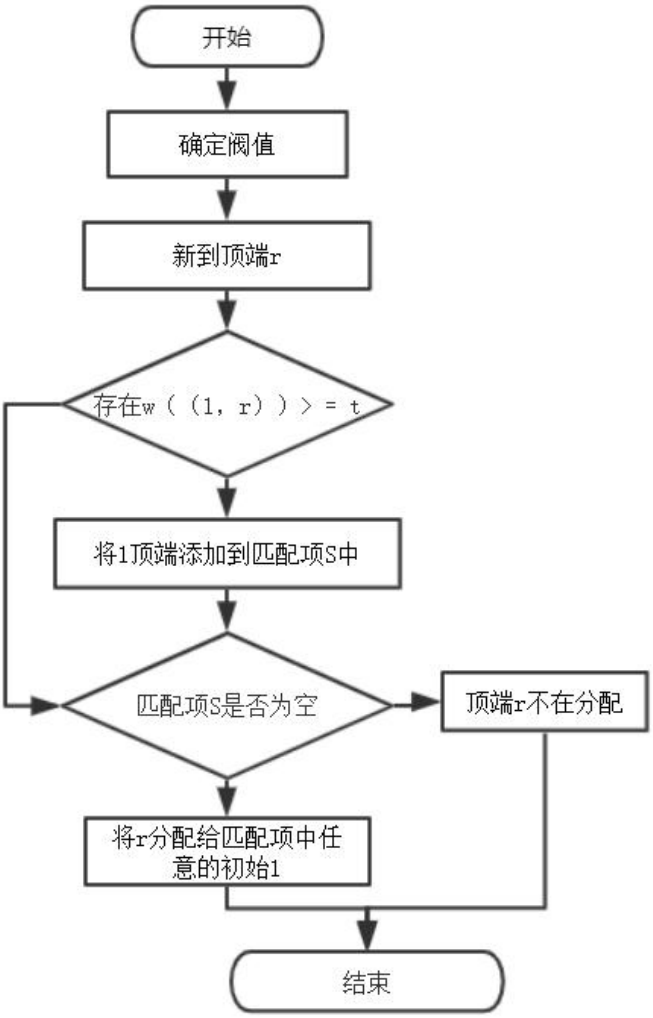
6.4.1 Greedy-RT 算法

Greedy-R 算法[2]在 OMWBM 问题的在线对抗性模型下具有最先进的竞争比为 1 2eln(1 Umax)，二部图加权在线到达的顶点只有一面。（UMAX 是最大权重）

Greedy-RT [2]的主要思想是首先在边的权重中随机选择一个阈值，然后对每个新到达的顶点（任务或者工人），任意边入射到其上的权重不小于所选阈值，如果存在这样的边，就将他添加到匹配项中，算法的具体实现如下：

Choose an integer κ randomly from the set $N = \{0, 1, 2, \ldots\}$ with probability $\Pr[\kappa = i] = p_i$;
Set $\tau = e^{\kappa}$;
while *a new vertex* $r \in R$ *arrives* **do**
  $S = \{\ell \mid \ell$ is r's unmatched neighbor in L and $w((\ell, r)) \geq \tau\}$;
  if $S = \emptyset$ **then**
    leave r unmatched forever;
  else
    match r to an arbitrary vertex in S;
  end
end

Greedy-RT 算法的流程图如下：



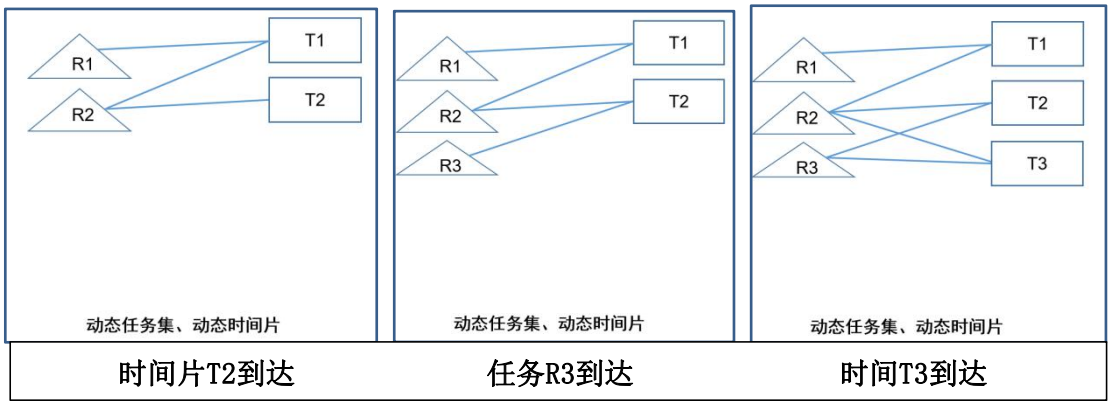
7 双边动态二分图匹配

7.1 动态任务集、动态时间片

任务集、时间片集、学习效率都会根据每名学生的状态动态发生变化，任务和时间片未来的信息无法预知，如何对动态变化的任务与时间片进行分配，达到最好的学习效率（已分配的任务—时间片匹配对不再改变）。

7.2 双边动态加权二分图匹配的定义

根据 6.1 学习任务分配的定义，可以将其建模成双边动态加权二分图匹配问题，允许任务与时间片以任意顺序动态地出现在二分图的任意位置，一旦一项新任务出现，立即为该任务分配一个当前未被分配的时间片或令其等待后续的时间片来执行它，直至该任务截止时间为止，反之，对一个新出现的时间片亦然。此外，当一个分配被确定后，则分配结果不可更改。其算法性能既受制于任务与时间片所构成的二分图结构，又依赖于任务与时间片的抵达顺序。任务集、时间片集动态到来过程如下图所示：



7.3 匹配算法

7.3.1 Extended-Greedy-RT 算法

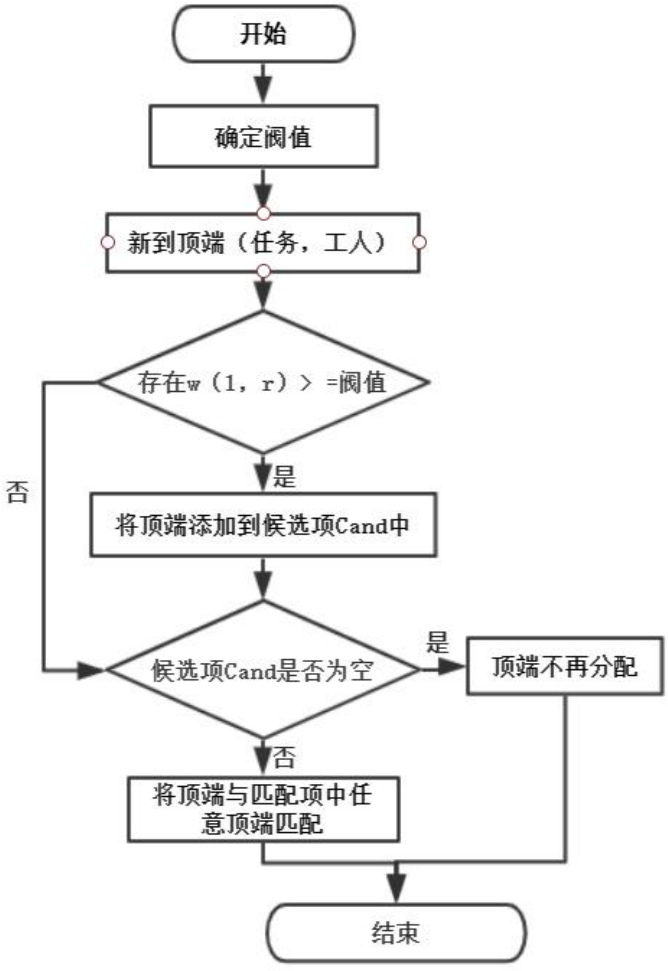
其框架与 Greedy-RT 相似，但加权二分图的顶点(即任务和时间片)是可以双方动态到达的。

Extended-Greedy-RT 根据估计的最大权重 Umax 随机选择边缘权重的阈值（ek），可以从平台上的历史记录中获悉。当新的顶点到达时（可能是任务或时间片），Extended-Greedy-RT 在其上加入一条边，其权重不小于阈值且满足匹配结果的所有约束条件。如果存在这样的边，就将他添加到匹配项中。算法的具体实现代码如下：

input : $T, W, U(.,.)$
output: A feasible allocation M

$\theta \leftarrow \lceil \ln(U_{max}+1) \rceil$;
$k \leftarrow$ randomly choosing an integer from $\{1, \cdots, \theta\}$ with probability $\frac{1}{\theta}$;
foreach *new arrival task or worker* v **do**
    $Cand \leftarrow \{\forall u | u$ is an unmatched neighbor of v such that $U(u,v) \geq e^k$ and it satisfies all constraints$\}$;
    if $Cand = \emptyset$ **then**
        continue;
    else
        $u^\star \leftarrow$ an arbitrary item is chosen from $Cand$;
        $M \leftarrow M \cup (u^\star, v)$;
return M

Extended-Greedy-RT 算法的流程图如下（与 Greedy-RT 算法的区别在于新到来的顶点既可以是任务也可以是时间片）：

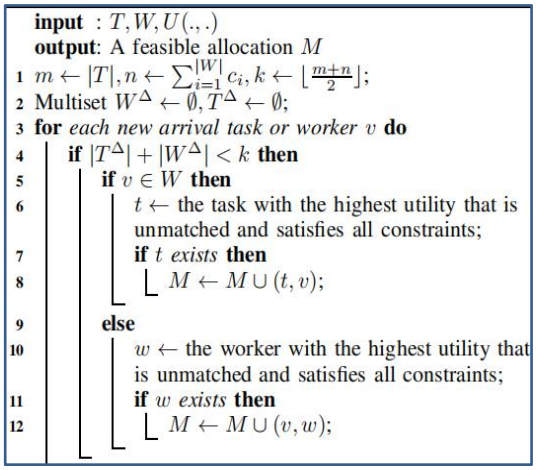


7.3.2 TGOA Algorithm

TGOA 算法是受到经典秘书问题方法的启发[3]，主要思想是首先划分所有顶点，每个顶点要么是任务，要么是时间片，根据他们到达的顺序采取不同的策略划分到两个相等的组中。 也就是说，顶点的前半部分首先到达，而顶点的后半部分之后到达。

注意，一段时间间隔下的任务和时间片总的数量是可以根据平台历史的记录估计的。

然后，对于任务和时间片的前半部分，TGOA 实施贪婪策略，以将每个新到达的任务（时间片）分配给效用最高且满足所有约束的相应时间片（任务）。对于任务和时间片的后半部分，当新任务或时间片到达时，为了减少前半部分任务和时间片陷入贪婪算法局部最优的可能性，TGOA 尝试执行全局最优匹配。算法的具体实现代码如下：



input : $T, W, U(.,.)$
output: A feasible allocation M

$m \leftarrow |T|, n \leftarrow \sum_{i=1}^{|W|} c_i, k \leftarrow \lfloor \frac{m+n}{2} \rfloor$;
Multiset $W^{\Delta} \leftarrow \emptyset, T^{\Delta} \leftarrow \emptyset$;
for *each new arrival task or worker* v **do**
  if $|T^{\Delta}| + |W^{\Delta}| < k$ **then**
    if $v \in W$ **then**
      $t \leftarrow$ the task with the highest utility that is unmatched and satisfies all constraints;
      if t *exists* **then**
        $M \leftarrow M \cup (t, v)$;
    else
      $w \leftarrow$ the worker with the highest utility that is unmatched and satisfies all constraints;
      if w *exists* **then**
        $M \leftarrow M \cup (v, w)$;
  else
    $M_v \leftarrow Hungarian(T^{\Delta} \cup W^{\Delta} \cup \{v\})$;
    if v *is matched in* M_v **then**
      if $v \in W$ **then**
        $t \leftarrow$ the task assigned to v in M_v;
        if t *is unmatched in* M *and satisfies all constraints* **then**
          $M \leftarrow M \cup (t, v)$;
      else
        $w \leftarrow$ the worker assigned v in M_v;
        if w *is unmatched in* M *and satisfies all constraints* **then**
          $M \leftarrow M \cup (v, w)$;
    if $v \in W$ **then**
      $W^{\Delta} \leftarrow W^{\Delta} \cup v$;
    else
      $T^{\Delta} \leftarrow T^{\Delta} \cup v$;
return M

7.3.3 TGOA-Greedy Algorithm

尽管 TGOA 算法在顶点后半部分采用了最佳策略以确保更高分配结果的质量，但它效率不够高，因为它花费三倍的时间复杂度去为每个后半部分节点找当前全局最优匹配。因此，我们通过替换带有贪心策略的最佳匈牙利算法来提高 TGOA 的效率，使得竞争比率略低。

TGOA-Greedy 算法的算法类似于 TGOA Algorithm 算法，具体地说，我们将匈牙利算法替换为贪心算法找到假设的匹配。 在到达的下半部分顶点中，我们迭代地找到它们之间具有最大效率值且满足所有约束的不匹配边，并将该边添加到匹配项中。算法的具体实现代码如下：

Algorithm 3: Greedy-Match

input : A baipartite graph
output: An offline allocation M_v^{Greedy}

$M_v^{Greedy} \leftarrow \emptyset$;
while *True* **do**
  $(t, w) \leftarrow$ the task-worker pair with the highest utility that is unmatched and satisfies all constraints;
  if (t, w) *exists* **then**
    $M_v^{Greedy} \leftarrow M_v^{Greedy} \cup (t, w)$;
  else
    break;
return M_v^{Greedy}

7.3.4 OffPSP Algorithm

OffPSP 算法每次匹配仍然选择最划算的一对。每个 t 时间片有一个调度表且有一个总的开销上限。首先做一个可能对上限不可行的伪分配，如果是第一次超过时间片的总开销上限，则允许进行伪分配。在最后一步，判断伪分配是否可行，检查了每一个不可行的时间片调度表，对不可行的调度表进行处理：将最后一个工作与调度表其它工作比较，保留一个更大的效率。具体实现代码如下：

$E' \leftarrow E$;
$i \leftarrow 1$;
foreach $t \in \{1, \ldots, T\}$ **do**
  $S_0^t \leftarrow \emptyset$;
$A_0 \leftarrow (S_0^1, \ldots, S_0^T)$;
while $E' \neq \emptyset$ **do**
  $(j_i, t) \leftarrow \arg\max_{(j,t) \in E'} \frac{U(S_{i-1}^t \cup \{j\}) - U(S_{i-1}^t)}{c_j}$;
  $S_i^t \leftarrow S_{i-1}^t \cup \{j_i\}$;
  $E' \leftarrow E' \backslash \delta(j_i)$;
  if $c(S_i^t) > C^t$ **then**
    $E' \leftarrow E' \backslash \delta(t)$;
  $i \leftarrow i + 1$;
foreach $t \in \{1, \ldots, T\}$ **do**
  if $c(S_i^t) > C^t$ **then**
    Let j be the last job scheduled into period t;
    if $U(\{j\}) > U(S_t^i \backslash \{j\})$ **then**
      $S_t^i \leftarrow \{j\}$;
    else
      $S_t^i \leftarrow S_t^i \backslash \{j\}$;
return $A \leftarrow (S_i^1, \ldots, S_i^T)$;

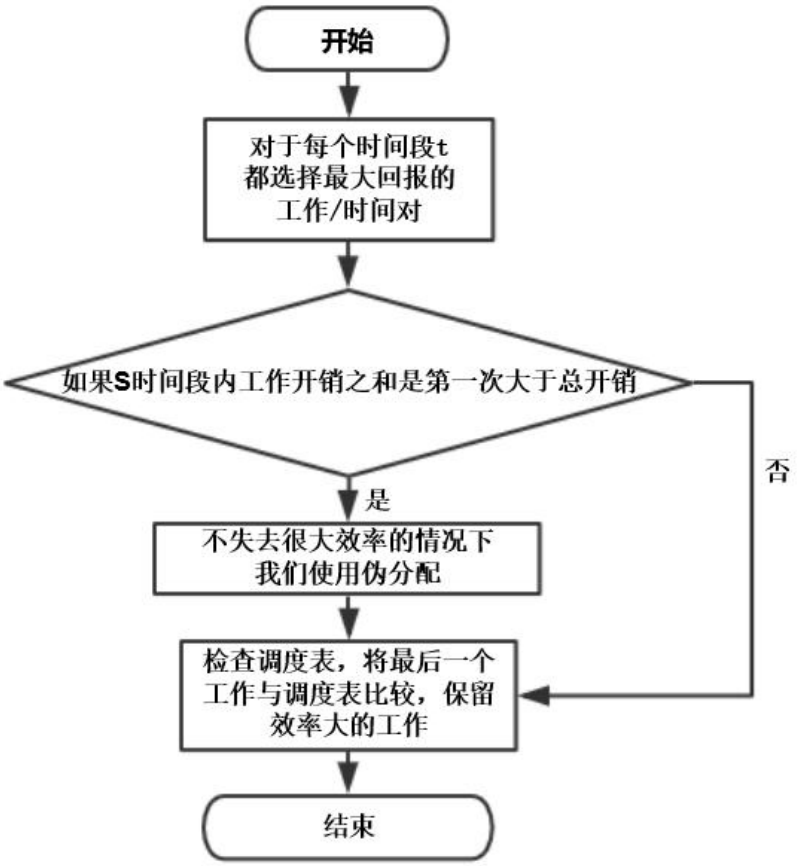
步骤：

1、仍然选择最大回报的工作/时间对 （lines 6-12）

2、但进行伪分配，如果是第一次大于 C^t，因此效率值会较大增加（lines 14）

3、在不失去很大效率的情况下，我们使用伪分配（lines 16-17）

4、最后，检查每个不可行的调度表，将最后一个工作与调度表其它工作比较，保留一个更大的效率（lines 18-19）



8 总结与分析

8.1 静态、单边动态、双边动态最大加权二分图匹配对比

静态二分图匹配的任务集和时间片集在分配过程中是已知不变的，虽然实现简单但是不符合现实场景，所使用的算法也是已经非常成熟，时空复杂度也有很好的改进；单边动态二分图虽然有一边点集是可以动态的，但仍然不是很符合现实场景；双边动态二分图在分配过程中两边顶点是动态到来的，符合现实场景但实现起来非常困难，且时空开销非常大。如下表所示：

	任务集	时间片集	是否符合现实	常用算法	竞争比率	模型
静态最大加权二分图匹配	不变	不变	否	KM算法	—	—
				Ford-Fulkerson算法		
单边动态最大加权二分图匹配	不变	变化	否	Greedy-RT算法	$\frac{1}{2e\ln(1+U_{max})}$	adversarial model
	变化	不变	否			
双边动态最大加权二部图匹配	变化	变化	是	Extended-Greedy-RT	$\frac{1}{2e\lceil\ln(U_{max}+1)\rceil}$	adversarial model
				TGOA Algorithm	1/4	Randomized model
				TGOA-Greedy Algorithm	1/8	Randomized model
				OffPSP Algorithm	—	—

解释：在线算法中竞争比率（CR）用于评估其效果，衡量在线算法与离线模型的最佳结果。

8.2 分析与评价

在真实的学习场景中，学生的时间表与学习任务都会随时发生变化，未来任务集与时间片集的信息根本无法预测，只能针对当前已存在的任务集与时间片集进行最优匹配处理，若时间片或任务后续发生动态变化，再针对新的任务集与时间片集进行处理（规定已分配的任务与时间片不能更改了）。因此双边动态最大加权二部图匹配是最符合真实的学生学习的场景。

9 未来工作

从分析所有算法发现，所有的新算法思想都是基于最基础的贪心算法和匈牙利算法上进行改进、提高从得出来的。现有的算法要么对分配效率进行改进要么对时空复杂对进行改进，但一种既具有高效的分配效率又能很好的解决时空开销问题的算法还值得进一步研究。

后面提出的四种算法都是用于解决在线任务分配及时空众包任务分配场景中，要完全适应于我们的学习任务分配场景中，可能需要再考虑增加新的约束条件以及剥除关于时间、空间等不相干问题。

9 参考文献

[1] Tong Y, She J, Ding B, et al. Online mobile micro-task allocation in spatial crowdsourcing[C]//2016 IEEE 32Nd international conference on data engineering (ICDE). IEEE, 2016: 49-60.

[2] H. Ting and X. Xiang, “Near optimal algorithms for online maximum edge-weighted b-matching and two-sided vertex-weighted b-matching,” Theoretical Computer Science, 2015.

[3] T. S. Ferguson, “Who solved the secretary problem?” Statistical science,1989.

[4] Deng D, Shahabi C, Zhu L. Task matching and scheduling for multiple workers in spatial crowdsourcing[C]//Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems. ACM, 2015: 21.

[5] H. To, G. Ghinita, and C. Shahabi, “A framework for protecting worker location privacy in spatial crowdsourcing,” PVLDB 2014.

[6] Song T, Tong Y, Wang L, et al. Trichromatic online matching in real-time spatial crowdsourcing[C]//2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, 2017: 1009-1020.

[7] R. E. Burkard, M. Dell’ Amico, and S. Martello, Assignment Problems,Revised Reprint, 2009.

[8] To, Hien. "Task Assignment in Spatial Crowdsourcing: Challenges and Approaches." Proceedings of the 3rd ACM SIGSPATIAL PhD Symposium (2016): 1-4. Web.

[9] R. M. Karp, U. V. Vazirani, and V. V. Vazirani, “An optimal algorithm for on-line bipartite matching,” in STOC 1990.

[10] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani, “Adwords and generalized online matching,” Journal of the ACM.

[11] H. Ting and X. Xiang, “Near optimal algorithms for online maximum edge-weighted b-matching and two-sided vertex-weighted b-matching,” Theoretical Computer Science, 2015.

[12] C.-J. Ho and J. W. Vaughan, “Online task assignment in crowdsourcing markets,” in AAAI 2012.

[13] U. ul Hassan and E. Curry, “A multi-armed bandit approach to online spatial task assignment,” in UIC 2014.

[14] A. Mehta, “Online matching and ad allocation,” Theoretical Computer Science, 2012.

[15] Wang Y, Tong Y, Long C, et al. Adaptive Dynamic Bipartite Graph Matching: A Reinforcement Learning Approach[C]//2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 2019: 1478-1489.

[16] Amador S, Okamoto S, Zivan R. Dynamic multi-agent task allocation with spatial and temporal constraints[C]//Twenty-Eighth AAAI Conference on Artificial Intelligence. 2014.

[17] Mehta A, Saberi A, Vazirani U, et al. Adwords and generalized on-line matching[C]//46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05). IEEE, 2005: 264-273.

[18] Aggarwal G, Goel G, Karande C, et al. Online vertex-weighted bipartite matching and single-bid budgeted allocations[C]//Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, 2011: 1253-1264.

[19] Wang L, Tong Y, Hu C, et al. Procrastination-Aware Scheduling: A Bipartite Graph Perspective[C]//2019 IEEE 35th International Conference on Data Engineering (ICDE). IEEE, 2019: 1650-1653.

[20] Zheng Y, Wang J, Li G, et al. QASCA: A quality-aware task assignment system for crowdsourcing applications[C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015: 1031-1046.

[21] Tong Y, Wang L, Zhou Z, et al. Flexible online task assignment in real-time spatial data[J]. Proceedings of the VLDB Endowment, 2017, 10(11): 1334-1345.

[22] Wicke, Drew, Luke, Sean, Axtell, Robert, Menasce, Daniel, and Simon, Robert. Bounty Hunting: A Dynamic Multiagent Task Allocation Mechanism (2018): ProQuest Dissertations and Theses. Web.

[23] Hoong Chuin Lau, and Lei Zhang. "Task Allocation via Multi-agent Coalition Formation: Taxonomy, Algorithms and Complexity." Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence (2003): 346-50. Web.

[24] L. Kazemi, C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In SIGSPATIAL/GIS, pages 189-198, 2012.

[25] H. To, C. Shahabi, L. Kazemi. A server-assigned spatial crowdsourcing framework. TSAS,
1(1):2, 2015.

[26] L. U, M. Yiu, K. Mouratidis, N. Mamoulis. Capacity constrained assignment in spatial
databases. In SIGMOD, pages 15-28, 2008.

[27] L. U, K. Mouratidis, M. Yiu, N. Mamoulis. Optimal matching between spatial datasets under capacity constraints. TODS 35(2): 9:1-9:44, 2010.

[28] L. U, N. Mamoulis, K. Mouratidis: A fair assignment algorithm for multiple reference queries.PVLDB, 2(1): 1054-1065, 2009.

[29] C. Long, R. Wong, P. S. Yu, M. Jiang. On optimal worst-case matching. In SIGMOD, pages 845-856, 2013.

[30] L. Kazemi, C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing.GIS 2012.

[31] R. C.-W. Wong, Y. Tao, A. W.-C. Fu, and X. Xiao, “On efficient spatial matching,” in VLDB 2007.

[32] Zheng Y, Wang J, Li G, et al. QASCA: A quality-aware task assignment system for crowdsourcing applications[C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015: 1031-1046.

[33] Tong Y, Wang L, Zhou Z, et al. Flexible online task assignment in real-time spatial data[J]. Proceedings of the VLDB Endowment, 2017, 10(11): 1334-1345.

[34] Wicke, Drew, Luke, Sean, Axtell, Robert, Menasce, Daniel, and Simon, Robert. Bounty Hunting: A Dynamic Multiagent Task Allocation Mechanism (2018): ProQuest Dissertations and Theses. Web.

[35] Hoong Chuin Lau, and Lei Zhang. "Task Allocation via Multi-agent Coalition Formation: Taxonomy, Algorithms and Complexity." Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence (2003): 346-50. Web.

[36] L. Kazemi, C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing. In SIGSPATIAL/GIS, pages 189-198, 2012.

[37] H. To, C. Shahabi, L. Kazemi. A server-assigned spatial crowdsourcing framework. TSAS,1(1):2, 2015.

[38] L. U, M. Yiu, K. Mouratidis, N. Mamoulis. Capacity constrained assignment in spatial databases. In SIGMOD, pages 15-28, 2008.

[39] L. U, K. Mouratidis, M. Yiu, N. Mamoulis. Optimal matching between spatial datasets under capacity constraints. TODS 35(2): 9:1-9:44, 2010.

[40] L. U, N. Mamoulis, K. Mouratidis: A fair assignment algorithm for multiple reference queries.PVLDB, 2(1): 1054-1065, 2009.

[41] C. Long, R. Wong, P. S. Yu, M. Jiang. On optimal worst-case matching. In SIGMOD, pages 845-856, 2013.

[42] L. Kazemi, C. Shahabi. Geocrowd: enabling query answering with spatial crowdsourcing.GIS 2012. H. To, C. Shahabi, L. Kazemi: A server-assigned spatial crowdsourcing framework. TSAS 2015