第42卷 第6期 西 安 交 通 大 学 学 报 Vol.42 №6
2008年6月 JOURNAL OF XI′AN JIAOTONG UNIVERSITY Jun. 2008

# 基于思维进化的集群作业调度方法研究

薛正华，刘伟哲，董小社，伍卫国
（西安交通大学计算机科学与技术系，710049，西安）

**摘要**：为了减少不准确的时间估计对作业调度的影响，提出了动态预约调度模式.在该模式中，预约队列中的作业与被预约资源是松耦合的，可根据作业完成情况进行重映射，以减少作业提前完成而产生的资源碎片.基于动态预约模式，将思维进化计算引入到作业调度中，以各种回填算法的调度结果作为初始群体，通过趋同和异化操作，使群体不断向最优解进化，从而产生更优的调度方案.基于真实作业集的仿真结果表明，所提算法的作业平均的等待、延迟时间比最优的基于回填的组合算法分别下降了68.5%和66.9%.

**关键词**：服务器集群；作业调度；思维进化计算
**中图分类号**：TP391.9 **文献标志码**：A **文章编号**：0253-987X(2008)06-0651-04

# Research on Mind Evolutionary Computation Based Job Scheduling for Server Clusters

XUE Zhenghua, LIU Weizhe, DONG Xiaoshe, WU Weiguo
(Department of Computer Science and Technology, Xi′an Jiaotong University, Xi′an 710049, China)

**Abstract**: A dynamic reservation based on scheduling method is proposed to reduce the disadvantage caused by inaccurate estimation on job run time. The method combines the reservation jobs and the computing resources in loose couple, and it enables the scheduler to recombine the reservation jobs and resources to reduce the idle resources caused by jobs completed in advance. Based on the dynamic reservation mode, the mind evolutionary computation (MEC) based algorithm is introduced into the scheduler. The algorithm sets the result obtained from the backfill algorithms as the initial population, and it can achieve better scheduling result by adopting assimilation and dissimilation operations. Simulations based on a real workload show that the proposed method is valid. Compared with the backfill algorithm, the proposed algorithm decreases both the average job wait time and the average slowdown by 68.5% and 66.9%, respectively.

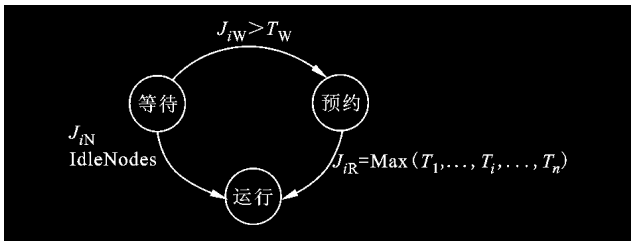
**Keywords**: server cluster; job scheduling; mind evolutionary computation

作业调度是高性能计算领域的一个重要研究内容，其在一定程度上属于NP类的组合优化问题.回填算法是目前广为采用的一种调度算法，集群作业调度算法多属于回填的改进算法，如动态调整回填深度法[1]、前瞻回填法[2]、紧密填充法[3]、放宽回填要求法[4]等，它们在某种程度上改进了回填算法，但回填算法本身是一种基于经验规则的算法，难以保证较高的调度质量.此外，现有的集群调度系统一般是通过用户提供的作业预估运行时间来分配资源的，而这种预估时间往往不够准确，从而影响调度质量.基于此，本文提出了一种动态预约调度模式，以减少不准确的预估时间对调度的影响，同时将思维进化计算(MEC)优化方法引入到作业调度中，利用MEC趋同和异化的性质获得较回填算法更优的调度效果.

收稿日期：2007-09-20. 作者简介：薛正华(1978—)，男，博士生；董小社(联系人)，男，教授，博士生导师. 基金项目：国家自然科学基金资助项目(60773118)；国家高技术研究发展计划资助项目(2004AA111110，2006AA01A109).

## 1 动态预约模式

作业在调度系统中的3种状态转换过程如图1所示.




$J_{iN}$:作业 $i$ 所需的节点数;$J_{iW}$:作业 $i$ 的等待时间;$I_N$:系统拥有的空闲节点数;$T_W$:系统设定的作业最长等待时间;$J_{iR}$:处于预约状态的作业 $i$ 转为运行状态的时刻;$T_i$:节点 $i$ 成为空闲的时刻

图1 作业状态转换

作业 $i$ 被提交之后,处于等待状态.若 $J_{iN}\leqslant I_N$,则作业 $i$ 可转为运行状态,否则继续等待.若作业 $i$ 的等待时间满足 $J_{iW}>T_W$,则作业调度器为其预约资源,作业 $i$ 转为预约状态,且被预约的资源不再分配给其他作业.设调度器为作业 $i$ 预约的节点为 $N_1,N_2,\cdots,N_n$,则作业 $i$ 在时刻 $J_{iR}=\mathrm{Max}(T_1,\cdots,T_i,\cdots,T_n)$转为运行状态.

**定义1** 作业 $i$ 预约的前 $j$ 个节点在 $T_1$ 时刻空闲,而剩下节点要在 $T_2$ 时刻才能空闲,则称前 $j$ 个节点在$(T_2-T_1)$时间段内为预期资源碎片.

**定义2** 用户预估的作业完成时间为 $T_1$,实际的作业完成时间为 $T_2(T_2<T_1)$,则在$(T_1-T_2)$时间段内作业所占节点为未预期资源碎片.

现有的作业调度系统是在用户提交作业时根据设定的预估运行时间分配资源的,若用户作业超过了预定时间,则杀掉该作业.因此,用户指定的预估时间一般大于实际运行时间,由此增大了作业提前完成的概率,增多了未预期资源碎片.若产生的未预期资源碎片是预约的资源,则将导致预约作业提前执行.由于预约作业与资源的映射是固定的,所以系统只能从等待队列中选取合适的作业,于是预约作业就丧失了从未预约的未预期资源碎片中获利的机会,调度器也可能就丧失了更优的调度方案.基于此,本文提出了一种动态预约模式,其核心是预约作业与预约资源的映射是松耦合的,系统若出现未预约的未预期资源碎片,调度器将重新计算,以重建预约队列中作业与资源的映射关系,从而获得更优的调度效果.动态预约模式算法的描述如图2所示.

```
Input:等待作业队列WaitList;
      预约作业队列ResList;
      运行作业队列RunList;
      节点队列NodeList;
Begin
  While(NodeList中出现空闲资源)
    {
     If(空闲资源为预期资源碎片)
       利用基于回填的混合调度算法,
       从WaitList中查找合适的作业放在预期资源
       碎片上;
     If(空闲资源为未预期资源碎片)
      {
       If(未预期资源碎片已被预约)
            检查被预约资源对应的预约作业是否可执行;
        If(对应的预约作业可开始执行)
            将该作业置入ResList中并开始运行;
        Else
            利用基于回填的混合调度算法,
            从WaitList中查找合适的作业放在被预约的
            未预期资源碎片上;
       If(未预期资源碎片未被预约)
        {
         调度器根据系统设定的调度算法重新计算
         ResList和NodeList的映射关系;
         If(调度器计算出更优的映射方案)
            解散原有的映射关系,按新映射方案重建
            ResList和NodeList的映射关系;
         Else
            利用基于回填的混合调度算法,
            从WaitList中查找合适的作业并置入未被预约
            的未预期资源碎片上;
        }
      }
    }
Else
```

图2 动态预约模式算法

## 2 思维进化算法

MEC是一种模拟人类思维进化的新型算法,通过趋同向优胜者学习,通过异化创造新思维来克服自然进化算法计算时间过长和计算结果不可知的缺点.本文将各种回填组合算法的调度结果作为初始群体,并利用MEC对这些结果改进,进而生成更优的调度结果.

### 2.1 基本流程

用于集群作业调度的MEC算法的基本流程如图3所示,其中生成局部公告板和根据局部公告板生成新个体的过程属于趋同,生成全局公告板、淘汰非优胜群体和根据全局公告板生成新个体的过程属于异化.

### 2.2 编码方式

作业调度的最终结果是为每个作业分配一个开始的运行时间,因此作业调度结果可表示为

$$J=\{(j_1,t_1),\cdots,(j_i,t_i),\cdots,(j_n,t_n)\} \quad (1)$$

式中:$j_i$ 表示作业编号;$t_i$ 表示作业开始的运行时间.对于某个调度方案,若将作业按开始的运行时间进行排序,就可得到一个作业编号序列

$$S=\{(j_1,\cdots,j_i,\cdots,j_n\} \quad (2)$$

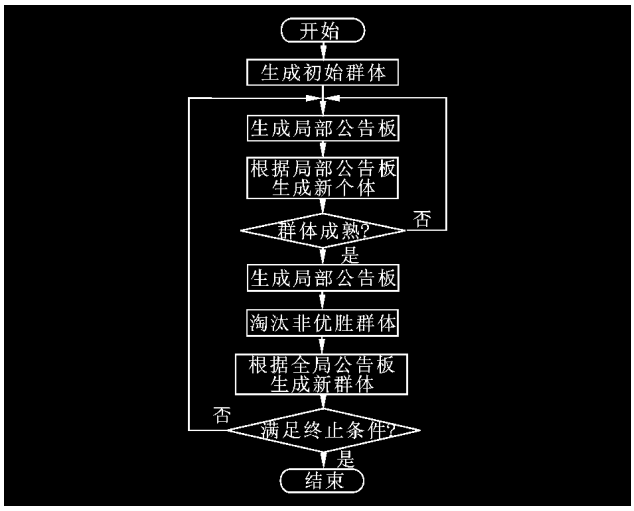



图 3　集群作业调度 MEC 算法流程

此序列即可作为此次调度方案的编码方式.

### 2.3　评价函数

作业调度的评价函数是一个向量函数，它的返回值可表示为$(f_1,\cdots,f_i,\cdots,f_m)$，其中 $f_i$ 为作业平均的等待、延迟和响应时间等. 为了便于比较函数的返回值，可采用加权法将向量标量化，其评价函数为

$$F=\sum w_i f_i \tag{3}$$

式中：$w_i$ 为参数权值. 由于每个 $f_i$ 的取值范围各不相同，所以需对其进行归一化处理. 评价函数的值为解的得分，调度的目标就是最大化解的得分.

### 2.4　初始化群体

在进行趋同和异化操作之前，先运行各种基于回填的组合调度算法，这些回填算法的解接近于最优解，再将它们作为初始群体可大大减少 MEC 算法的收敛时间. 考虑到不同参数回填得到的解有可能相同，因此需去除那些重复解. 为了使初始解不至于太少，还需随机生成一些解作为初始群体.

### 2.5　趋同处理

趋同是群体内的个体 $v$ 向优胜个体学习的过程，局部公告板则记录本群体优胜个体的特征信息. 在所有优胜个体中，若编码片段 $ij$ 出现的频度较高，就可认为含有编码片段 $ij$ 的个体是最优解的概率较大. 因此，可以先统计各个片段的出现概率，然后根据这些概率生成新一代个体.

**定义 3**　设一个群体中个体的最高得分为 $m$，则该群体中得分大于 $km$ $(k<1)$ 的个体为该群体的优胜个体，优胜个体的集合记为 $I_S$.

**定义 4**　在所有优胜个体的编码中，各编码片段出现次数的矩阵为 $\boldsymbol{B}=\{b_{ij}\}_{n\times n}$，其中 $n$ 为个体的编码长度，$b_{ij}$ 等于编码片段 $ij$ 在所有优胜个体的得分之和.

**定义 5**　在最优解的编码中，各个编码片段 $ij$ 出现的概率矩阵为 $\boldsymbol{C}=\{c_{ij}\}_{n\times n}$，其中 $n$ 为个体的编码长度，$c_{ij}$ 为编码片段 $ij$ 出现的概率.

在生成新一代个体之前，先计算频度矩阵，生成算法如图 4 所示.

```
Input:上一代的频度矩阵B_{t-1}, 优胜群体集合I_S
Outpt:频度矩阵B_i
Begin
    For each b_ij in B
        b_ij=wb_ij
        For each v in I_S
            If(i, j∈code(v))
                b_ij=score(v)+b_ij
End
```

$w$：上一代公告板的权值

图 4　频度矩阵生成算法

根据频度矩阵可计算概率矩阵，其生成算法如图 5 所示.

```
Input:  B
Outpt:  C
Begin
    For each b_ij in B
        c_ij=b_ij/∑_j b_ij
        For each v c_ij in C
            If(c_ij=0)
                c_ij=q/n
            Else
                c_ij=c_ij(1-q)+q/n
```

图 5　概率矩阵生成算法

先选择起始编码 $i$，然后以概率 $c_{ij}$ 选择后续编码 $j$，接着以概率 $c_{jk}$ 选择后续编码 $k$，依次类推，直到选择 $n$ 个编码为止. 一旦选择了编码 $i$，就要修改概率矩阵，以使算法不再将 $i$ 作为后续编码.

在生成了新一代个体之后，计算新个体的得分，若连续几次新个体的最高得分没有改进，就终止趋同操作，否则减小随机化因子 $q$，继续生成新一代个体.

### 2.6　异化处理

在完成一次趋同操作之后，选出得分最高的 $S$ 个群体作为优胜群体，并废弃其余的 $\Gamma$ 个群体，然后根据优胜群体中得分较高的个体计算全局公告板，全局公告板的计算方法如同局部公告板，计算完成后，需重新生成 $\Gamma$ 个新群体. 在生成新群体时，应对每个群体用全局公告板指导生成若干个个体，此外还需随机生成其他个体，以避免算法过早陷入局

部最优.

异化结束后,算法判断是否结束计算,而本文设定的算法终止条件是,本代群体的最高得分较之上一代群体没有改进,且计算时间不超过设定的阈值(避免调度时间过长),计算终止,输出得分最高的个体解码,此时的算法将作为作业调度算法,否则继续进行趋同和异化操作.

## 3 仿真测试

### 3.1 仿真模型

本文使用Parallel Workloads Archive[5]提供的真实作业集SDSC DataStar作为仿真的作业集.为了比较同一作业集在不同到达密度下的调度效果,需为每一作业集定义平均负载参数,通过调整负载参数可以仿真作业的到达密度.

作业集的平均负载为

$$L = \sum P_i t_{eti} / P\Delta t \tag{4}$$

式中:$P_i$ 是作业 $i$ 所需的处理器数;$t_{eti}$ 是作业 $i$ 的估计运行时间;$P$ 是系统的总处理器数;$\Delta t$ 是最后一个作业与第一个作业的到达时间之差.具体的作业记录如表1所示.

表1 SDSC DataStar作业记录

| $P$ | 作业数 | $L$ | 开始时刻 | 结束时刻 |
|---|---|---|---|---|
| 1 664 | 15 932 | 0.71 | 2004-11-21 20:57:39 | 2005-02-28 20:43:51 |

回填算法有多种组合,常用的按队列排序方式的有8种:XFactor(作业的队列排序等于当前时间与作业的到达时间之差除以作业的估计运行时间),先来先服务,短作业优先,长作业优先,窄作业优先,宽作业优先,小作业优先和大作业优先.回填方式包括首次适应(FirstFit)和最合适(BestFit)2种,回填深度从1个到4个,因此本文将8×4×2种不同的回填组合算法作为MEC算法的初始群体.

在仿真模型中,MEC算法的优胜群体数设为2,非优胜群体数设为6,每个群体内的个体数设为32,当计算时间超过5 s时终止计算.

### 3.2 结果及分析

将作业平均的等待、延迟时间(bounded slowdown)[6]作为调度算法的评价指标,每次取调度结果最好的回填组合算法与MEC算法进行比较,结果如图6和图7所示.

基于原始的SDSC DataStar作业集($L$ = 0.71),最优的回填组合算法和基于动态预约MEC算法的平均等待时间别为7 314 s和2 306 s,平均延迟分别为17.8 s和5.9 s.显然,MEC的平均等待时间和延迟降低了68.5%和66.9%.图6和图7的测试结果还表明,MEC对系统的两项指标有所改进,改进的程度将随着负载的增大而加大.这是因为,随着负载的增加,单位时间内到达的作业数增多,这样算法组合的方式就越多,也就可能获得更好的调度质量.此外,动态预约模式改变了原有的固定映射模式,从而为MEC算法提供了更多的组合方式.

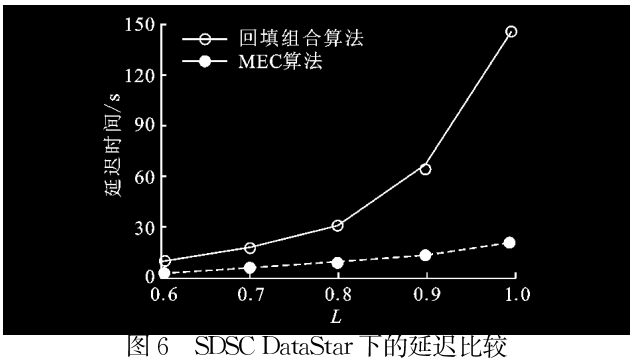



图6 SDSC DataStar下的延迟比较

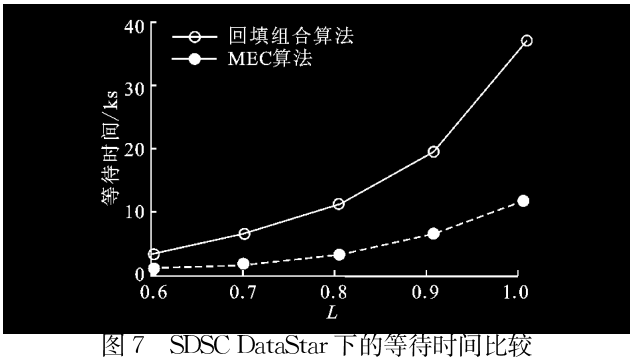



图7 SDSC DataStar下的等待时间比较

对MEC引入的时间开销进行了测试,MEC终止条件如2.6节所述.设定算法所耗时间不得超过30 s,那么不同等待队列长度(即等待作业数)下的算法最大时间开销如图8所示.从中看出,MEC算法所耗时间将随着等待作业队列长度的增加呈指数增长.

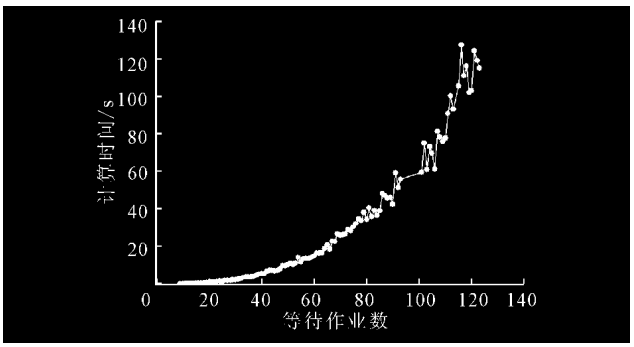



图8 MEC算法的时间开销

对 MEC 在 SDSC Datastar 样本下的计算时间进行了统计，其中 62.92%的 MEC 计算时间不超过 1 s，有 75.29%的不超过 2 s，有 85.01%的不超过 5 s，有 92.57%的不超过 10 s，相对于作业的运行时间来说，MEC 计算耗时在可接受的范围之内.

## 4　结　论

为了减少作业估计运行时间不准确给系统调度带来的影响，提出了一种动态预约调度模式，以减少作业提前完成而产生的资源碎片，保证预约作业优先完成. 将 MEC 算法引入到集群作业调度中，通过对现有调度策略的学习和改进，生成更优的调度方案. 在真实负载下的仿真测试结果表明，基于动态预约模式的 MEC 调度算法对系统作业的平均等待时间和平均延迟都有明显的改善，且优于回填算法，时间开销也比较小.

**参考文献：**

[1] SRINIVASAN S, KETTIMUTHU R, SUBRAMANI V, et al. Selective reservation strategies for backfill job scheduling [C]// 8th International Workshop on Job Scheduling Strategies for Parallel Processing. Berlin, Germany: Springer-Verlag, 2002:55-71.

[2] SHMUELI E, FEITELSON D. Backfilling with look ahead to optimize the performance of parallel job scheduling [C]// 9th International Workshop on Job Scheduling Strategies for Parallel Processing. Berlin, Germany: Springer-Verlag, 2003:228-251.

[3] LAWSON B, SMIRNI E. Multiple-queue backfilling scheduling with priorities and reservations for parallel systems [C]// 8th International Workshop on Job Scheduling Strategies for Parallel Processing. Berlin, Germany: Springer-Verlag, 2002:72-87.

[4] WARD W, MAHOOD C Jr, WEST J. Scheduling jobs on parallel systems using a relaxed backfill strategy [C]// 8th International Workshop on Job Scheduling Strategies for Parallel Processing. Berlin, Germany: Springer-Verlag, 2002: 88-102.

[5] DROR F. Parallel workloads archive [EB/OL]. [2007-09-20]. http://www.cs.huji.ac.il/labs/parallel/workload/index.html.

[6] XUE Zhenghua, DONG Xiaoshe, MA Siyuan et al. An energy-efficient management mechanism for large-scale server clusters [C]// Proceedings of the 2nd IEEE Asia-Pacific Services Computing. Piscataway, NJ, USA: IEEE, 2007:509-516.

（编辑　苗凌）