基于异构环境的Out-Tree任务图的调度算法

张建军 宋业新 旷 文
(海军工程大学理学院 武汉 430033)

摘 要 分布式应用程序的有效调度是异构计算系统中的一个关键问题。目前已有的Out-Tree任务图的调度算法大多基于同构环境而开发,未考虑处理机的异构性,导致调度的效率较低。针对异构计算环境,提出一个基于列表和任务复制的Out-Tree任务图的静态启发式贪心调度算法,其时间复杂度为$O(hv^2p)$,其中h、v和p分别表示任务图的高度、任务个数和调度使用的处理机个数。实验结果表明,相比其他算法,该算法能提供调度长度较短、处理机使用较少的有效调度,其应用性更强。

关键词 任务调度,Out-Tree任务图,异构性,任务复制,列表调度,调度长度
中图法分类号 TP316 **文献标识码** A

Heterogeneity Based Algorithm for Scheduling Out-Tree Task Graphs

ZHANG Jian-jun SONG Ye-xin KUANG Wen
(College of Science, Naval University of Engineering, Wuhan 430033, China)

Abstract Effective scheduling of a distributed application is one of the critical issues in heterogeneous computing systems. Many previous algorithms are developed based on homogeneous systems and neglect the heterogeneity of processors, which results in poor schedule efficiency. This paper proposed a heterogeneity, list and task duplication based static heuristics greedy algorithm for scheduling Out-Tree task graphs, the time complexity of which is $O(hv^2p)$ where h, v and p are the height of the DAG, the number of tasks in the task graph and the number of processors required respectively. The experimental results show that the proposed algorithm produces an efficient schedule which has shorter schedule length and less number of used processors in comparison with other related algorithms and so is more practical.

Keywords Task scheduling, Out-Tree task graph, Heterogeneity, Task duplication, List scheduling, Schedule length

1 引言

异构计算系统(Heterogeneous Computing System, HCS)由高速网络连接的具有不同计算速度的高性能计算机组成,旨在提供对计算密集型应用的高速处理。应用程序任务的有效调度是并行分布式系统获得高性能的关键因素。任务调度的目标是将任务映射到异构处理机并恰当安排它们的执行顺序,使任务之间的优先权关系得以满足的同时获得最小的调度长度。任务调度可以在编译时或运行时执行。当一个应用程序的主要特征,如各任务的执行时间及任务间的数据依赖关系均已预先得知,该问题即可用静态模型表示。HCS中的静态任务调度问题是NP完全问题,目前,对于同构系统和异构系统,已经提出多种启发式任务调度算法[1-10],这些算法基本可分为几类,主要的有基于列表的算法、基于聚簇的算法、基于任务复制的算法和以遗传算法为代表的智能算法等。其中,基于列表的调度算法常常能获得好的调度质量和性能。

在分布式环境中,一个任务调度系统模型可由应用程序、目标计算环境以及调度的性能准则组成。一个并行应用程序可用加权有向无环图(DAG)来描述。通常,DAG由四元组$G=(V,E,w,c)$来表示,其中V是由$v=|V|$个任务组成的任务集;E是由$e=|E|$条任务之间的通信边构成的边的集合;$w(n_i)$代表任务n_i($n_i\in V$)的计算量;由任务n_i传输到任务n_j的通信量用$c(n_i,n_j)$表示,记为c_{ij}。对任务n_i,$pred(n_i)=\{n_k|e(n_k,n_i)\in E\}$表示其父任务集合,$succ(n_i)=\{n_k|e(n_i,n_k)\in E\}$表示其子任务集合。没有任何父节点的任务$n_i$称为初始任务;而没有任何子任务的任务$n_i$称为叶子任务。对异构计算系统,通常假定处理机之间不存在通信竞争、计算和通信可重叠进行、给定应用程序的任务的执行是非抢占式的。

本文考虑HCS中Out-Tree任务图的调度问题。许多并行应用呈现Out-Tree这一常用模型结构,如图1所示。该种任务图的特点在于除了根节点之外,每个节点都只有一个父节点。HCS中该种任务图的调度是并行计算中基本而重要的问题,解决好此类任务图的调度问题,对实现并行语言的编译器、提高并行计算的性能具有非常重要的作用。

到稿日期:2012-06-05 返修日期:2012-09-20 本文受国家自然科学基金项目(71171198),海军工程大学科学基金项目(HGDJJ05005)资助。
张建军(1966-),男,博士,副教授,主要研究方向为并行计算、数据挖掘,E-mail:wahh0912@163.com;宋业新(1968-),男,博士,教授,主要研究方向为对策与决策、模糊系统;旷 文(1987-),男,硕士生,主要研究方向为算法分析与设计。

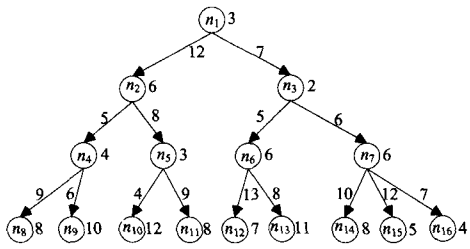

图 1　Out-Tree 任务图实例

目前，已有一些基于同构环境的 Out-Tree 任务图的调度问题的工作[4-6,8,10]，如 DCP[4] 算法、TDS[5] 算法和 TSA_OT[6] 算法等。其中，基于关键路径的 DCP 算法未采用任务复制的策略，产生的调度长度不如 TDS 算法的好；而基于任务复制的 TDS 算法在较强的条件下能获得最优调度，但调度 Out-Tree 任务图时会使用较多的处理器；而 TSA_OT 算法的复杂度为 $O(ev^2)$，该算法利用任务复制将通信开销化为零，且所采用的处理器个数小于或(在最坏情况下)等于 TDS 算法所采用的处理器个数，但是该算法只适用于独占的同构计算系统，在应用中常常不能取得期望的效果。实际计算环境往往有异构性，如果仍然采取基于同构环境的某些调度策略，由于处理机能力的差异可能会导致不恰当的负载平衡，从而导致 HCS 性能的下降。因此，为了更好地挖掘实际应用中 HCS 的并行处理的潜能，有必要深入研究 HCS 中的任务调度问题。

针对异构环境，从 Out-Tree 任务图的实际特征出发，本文采用表调度和任务复制相结合的调度机制，提出一种新的贪心调度算法(即 List and Task Duplication based Greedy Scheduling Algorithm 算法，简称 LTDGS_OT)，以提高调度的综合性能。试验结果表明，该算法能产生比 TDS 和 TSA_OT 等算法更好的调度结果，实用性较强。

2　LTDGS 算法

2.1　算法描述

LTDGS_OT 算法基于一些参数。不失一般性，设处理机集合为 $P\_SET=\{p_1,p_2,\cdots,p_m\}$，并假设各个处理机已按执行速度的降序排序，其速度依次记为 $k(1)$、$k(2)$、…、$k(m)$；令 $k(1)=1$，并且 $k(1)\geqslant k(2)\geqslant k(3)\geqslant\cdots\geqslant k(m)$。

以参数 $est(n_i)$ 和 $eft(n_i)$ 分别表示任务 n_i 的最早执行开始和最早执行完成时间，对初始任务，有 $est(n_{entry})=0$，$eft(n_{entry})=w(n_{entry})$。

对 Out-Tree 任务图中的其它任务，可以采用递归的方式从上而下计算对应的 est 和 eft 值，如下式所示：

$$est(n_i)=eft(pred(n_i)),eft(n_i)=est(n_i)+w(n_i)$$

对给定的 Out-Tree 任务图，定义其关键路径(Critical Path，CP)为同构环境下的具有最大执行时间的路径。如对图 1，其 CP 由任务 n_1、n_2、n_4 和 n_{10} 组成，长度为 28。

定义 1　对给定的 Out-Tree 任务图和 HCS，在某一调度步，考虑某候选任务(简称为当前任务)的调度时，参数 $len(p_j)$ 表示处理机 p_j 当前总的执行时间，即当前(该调度步之前)已调度至该处理机的所有任务的执行时间之和，其表达式如式(1)所示。式(1)中，参数 T_j 为当前已调度至该处理机的任务集合。相应地，在某一调度步，调度算法的当前调度长度 $T\text{-}len$ 规定为集合 $\{len(p_j)\mid p_j\in P\_SET\}$ 中的最大元素，如式(2)所示，P_SET 为当前已使用的处理机的集合。

$$len(p_j)=\sum_{n_i\in T_j}(w(n_i)/k(j)) \tag{1}$$

$$T\text{-}len=\max(len(p_j))\ (p_j\in P\_SET) \tag{2}$$

因此，LTDGS_OT 算法的调度长度即 *makespan* 恰好就是参数 $T\text{-}len$ 的终值。

算法分成两个主要阶段。首先是各叶子节点的优先权决定阶段。算法采用 top-down 方式计算所有叶子的 eft 属性值，并将它们按 eft 值的降序进行排序，将结果存入列表 *l_task* 中。

第二个阶段为处理机的选择阶段。算法首先将具有最大 eft 值的叶子节点及其所有祖先节点(即 CP 上的所有节点)调度至第一个处理机中；对列表 *l_task* 中其它叶子节点选择合适的处理机时，再采用如下的贪心策略。

(1) 若将当前叶子节点及其祖先节点不重复地调度到最快的处理机 p_1，对应的总执行时间即新的 $len(p_1)$ 值不大于算法的当前调度长度，即完成该叶子调度前的 $T\text{-}len$ 值，则将该叶子节点及其祖先节点不重复地分配到 p_1 中并依序执行。

(2) 若(1)中条件不成立，则进一步讨论：考虑新增处理机，从已有处理机 p_1 开始考察：如果将当前叶子节点及其祖先节点不重复地调度到当前考察的已使用的处理机 p_j，对应的总执行时间即新的 $len(p_j)$ 值小于或等于将其分配至新的处理机对应的执行时间，则立即将该叶子节点及其祖先节点不重复地调度到该已使用的处理机 p_j 中；否则，再考虑下一已用处理机 p_{j+1}，直至考虑完所有的已用处理机，在上述意义下决定是否可以将当前叶子节点调度至已用处理机。

(3) 若所有已用的处理机均不满足(1)和(2)中的条件，则将当前叶子节点及其祖先节点不重复地调度到新处理机中。

应该注意，算法在采用列表调度策略对当前考察的叶子节点进行分配时，采用复制策略时并不是盲目地复制其所有祖先任务，而是检查某些祖先是否已调度至该处理机，不去简单地重复复制它们，而是采用不重复地进行复制的方式。

采用上述策略，算法遍历完表 *l_task* 中的所有叶子节点，直至算法结束。LTDGS_OT 算法如图 2 所示，不妨假设叶子节点的个数为 m。

算法 1　LTDGS-OT(V,E,w,c)

```
Begin:
  计算各叶子任务的参数 eft(n_i)，按其降序对叶子排序，不妨记结果
  为 n_i1，n_i2，…，n_im，并将其依次存入链表 l_task 中；
  j=1；k=1；
  将叶子任务 n_ik 从 l_task 中取出；
  将 n_ik 及其祖先节点调度至处理机 p_j 中，计算 len (p_j)，并将 n_ik 及其
  祖先节点依次存入堆栈 s_j 中，转存至队列 q_j，将 p_j 存入集合 P_
  SET；
  T-len=len(p_j)；
    while (l_task≠Φ)
  {
    k++；
    j++；
    假设将 n_ik 及其祖先节点调度至处理机 p_j 中，计算 len(p_j)，从 l_
    task 中取出 n_ik；
```

```
for (t=1;t<j;t++)
  {
  假设将 n_ik及其祖先节点调度至处理机 p_t 中,计算 len (p_t),令
  Temp= len(p_t);
  if (Temp<=T-len)
    len(p_t)= Temp;
    将 n_ik及其祖先节点调度至处理机 p_t,将 n_ik及其祖先节点依
    次存入堆栈 Ts_t 中,将 Ts_t 中与 q_t 不重复的节点转存至队列
    q_t 之后;
  j=j-1;
  break for;
  else if (Temp<=len(p_j))
    len(p_t)= Temp;
    将 n_ik及其祖先节点调度至处理机 p_t,将 n_ik及其祖先节点依
    次存入堆栈 Ts_t 中,将 Ts_t 中与 q_t 不重复的节点转存至队列
    q_t 之后;
    T-len=len(p_t);
    j=j-1;
    break for;
  if (t=j-1)
    T-len=len(p_j);
    将 n_ik及其祖先节点调度至处理机 p_j,将 n_ik及其祖先节点依
    次存入堆栈 s_j 中,转存至队列 q_j,并将 p_j 存入集合 P_SET;
    break for;
  }
}
makespan=T-len;
End
```

图 2　LTDGS_OT 算法的详细描述

2.2　时间复杂性

LTDGS_OT 算法首先遍历任务图中各任务，以计算各叶子节点的 *est* 和 *eft* 值并进行快速排序，其时间复杂度为 $O(v)+O(v\log v)$；然后算法考察所有的叶子，决定如何将当前叶子及其祖先节点加入到已使用的处理机或新处理机，所有的已用处理机都可能被检查，需要的总时间为 $O(vp)$；最后，将叶子节点及其祖先节点调度至对应处理机，最坏的情形是，其所有祖先节点均需被复制，该步对应的时间复杂度为 $O(hv)$。因此，本算法总的时间复杂度为 $O(hv^2p)$。

2.3　算法的有效性

LTDGS_OT 算法适用于任意 Out-Tree 任务图并生成有效的调度，有下列结果。

定理 1　对于给定的 Out-Tree 任务图和 HCS，LTDGS_OT 算法生成一个有效的调度，其调度长度即 *makespan*，为集合 $\{len(p_j)\mid p_j\in P\_SET\}$ 中的最大元素，其中 $len(p_j)$ 及 *P_SET* 分别为对应参数的终值，即

$$makespan=\max_{j\in P\_SET}\{len(p_j)\}$$

证明：结论从 LTDGS_OT 算法的描述即知。证毕。

3　算法性能分析

3.1　实例分析

为阐明 LTDGS_OT 算法的有效性，本节给出算法对任务图 1 调度的运行轨迹。本调度最多可能要用到 9 个处理机，所以令 $m=9$；并不妨假设 $k(1)=1$、$k(2)=1/2$、$k(3)=1/2$、$k(4)=1/3$、$k(5)=1/4$、$k(6)=1/5$、$k(7)=1/6$、$k(8)=1/7$ 及 $k(9)=1/7$。

第一阶段，计算各叶子的 *eft* 属性，如表 1 所列，表 *l_task* 中的节点依次为 n_{10}、n_9、n_{13}、n_8、n_{11}、n_{14}、n_{12}、n_{15} 及 n_{16}。

表 1　各叶子节点的最早完成时间

n_i	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}
eft	21	23	24	20	18	22	19	16	15

第二阶段，算法先将 n_{10} 及其所有祖先节点即关键路径上的任务调度至处理速度最快的处理机 p_1 中，这时，$T\text{-}len=len(p_1)=24$，再将 p_1 加入到集合 *P_SET* 中。

考察表 *l_task* 中的第二个任务 n_9，因 $len(p_2)=46$，且 $len(p_1)+(w(n_9)+w(n_4))/1=38>T\text{-}len$，$len(p_1)+(w(n_9)+w(n_4))/1=38<len(p_2)$，从而算法将 n_9 及其祖先节点 n_4 调度至处理机 p_1 中，更新 $len(p_1)=38$，$T\text{-}len=38$。

类似地，对于第 3 个叶子节点 n_{13}，由于 $len(p_2)=44$，且 $len(p_1)+(w(n_{13})+w(n_6)+w(n_3))/1=57>T\text{-}len$，$len(p_1)+(w(n_{13})+w(n_6)+w(n_3)/1=57>len(p_2)$，因此算法将 n_{13} 及其所有祖先节点调度至新处理机 p_2 中，更新 $len(p_2)=44$，$T\text{-}len=44$，并将 p_2 加入到集合 *P_SET* 中。

采用上述列表和任务复制相结合的调度策略，该算法在完成列表中后续叶子节点的调度过程中，在选中的处理机中全部或部分复制当前考察叶子的祖先节点（如有必要），当完成对叶子任务 n_8、n_{11}、n_{14}、n_{12}、n_{15} 及 n_{16} 的分配后，$l\_task=\Phi$，算法终止。其处理机分配及调度时间如图 3 所示，LTDGS_OT 算法仅使用了 5 个处理机，调度长度仅为 71。

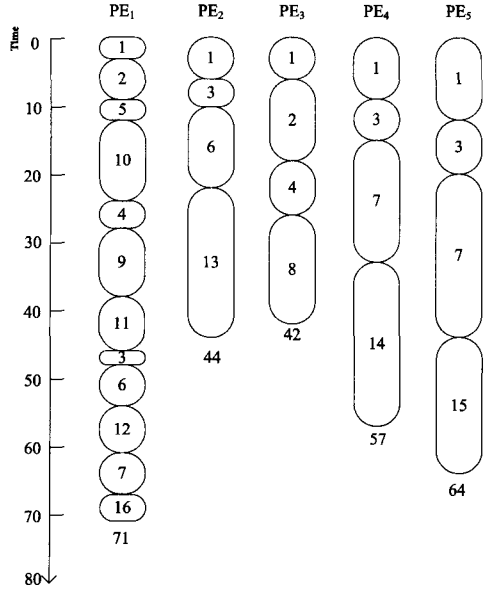

图 3　LTDGS_OT 算法调度图 1 的结果

3.2　性能分析

将 TSA_OT 和 TDS 算法运用于图 1，基本思想是：先按上述调度算法进行处理机分配，再根据上述异构环境的假定，改变原算法中各任务的开始执行时间和完成时间，得到的调度结果分别如图 4、图 5 所示。对运用本算法以及 TDS 和 TSA_OT 算法调度图 1 的调度性能指标进行比较，结果如表 2 所列。

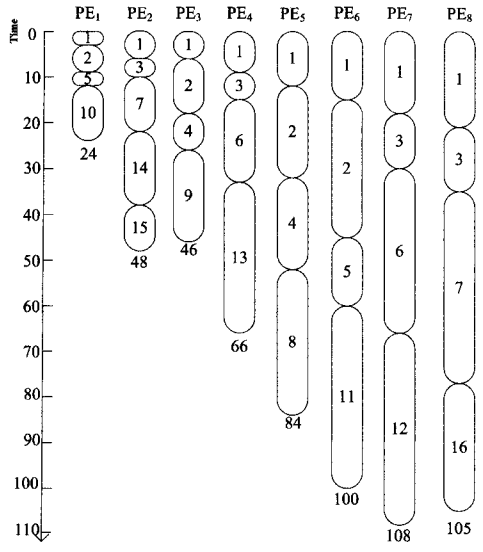

图 4　TSA_OT 算法调度图 1 的结果

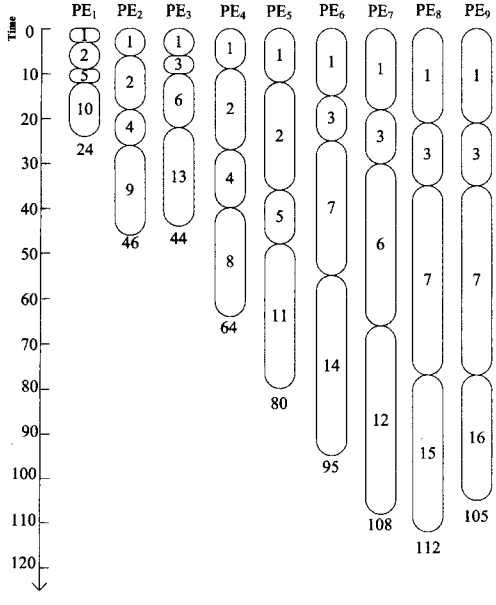

图 5　TDS 算法调度图 1 的结果

表 2　LTDGS_OT 算法与其他算法的比较

调度算法	调度长度	使用处理机数	时间复杂度
LTDGS_OT	71	5	$O(hv^2p)$
TDS	112	9	$O(v^2)$
TSA_OT	108	8	$O(ev^2)$

从图 3—图 5 可以看出，由于 Out-Tree 任务图的结构中没有 join 节点，因此 TDS 算法将不同的叶子节点调度至不同的处理机中，忽视了节约处理机，这里使用了 8 个处理机，且不能保证产生最优调度长度。TSA_OT 算法尽管在同构环境下能获得非常好的调度效果，但在 HCS 中使用了 7 个处理机，调度过程中虽也注重处理机之间的负载平衡，但算法并没有考虑处理机的异构性，因而在异构环境中产生的调度效果并不理想。相比之下，LTDGS_OT 算法的时间复杂度虽略有提高，但能产生较好的调度效果：其产生的调度长度仅为 71，比 TDS 短 36.6%、比 TSA_OT 短 34.3%，使用的处理机个数仅为 5 个，明显少于其它两个算法，且各处理机的负载相对最为平衡；而其它两个算法不仅调度长度较长，且在处理机的负载平衡方面显然效果欠佳。从实例来看，本算法产生的调度结果最佳。

为更全面地评估和比较 LTDGS_OT 算法、TSA_OT 算法和 TDS 算法，将随机生成的 Out-Tree 任务图作为测试这些算法的负载。首先生成 20 到 200 个节点，对应的计算量和任务间的通信量均在 1 至 20 之间，本测试中 CCR 取 1.3。Out-Tree 任务图采用相关的经典方法生成。仿真程序采用 Visual C，并采用 Windows XP，1G RAM 及 1.9GHz CPU 的个人计算机。用随机生成的 11 个 Out-Tree 任务图作为测试这些算法的负载，测试中对 HCS 的相关参数进行同样的随机假设。LTDGS_OT 算法、TSA_OT 算法和 TDS 算法的调度长度和所用处理机数的比较结果分别如图 6、图 7 所示。

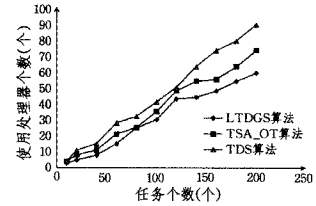

图 6　调度所用处理机数的比较结果

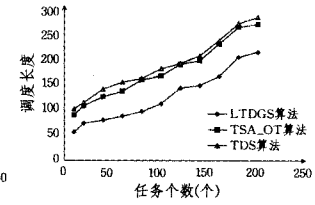

图 7　调度长度的比较结果

由图 6 和图 7 可以看出，相比其它两个算法，对于不同的任务个数，LTDGS_OT 算法使用的处理机数和调度长度均为最小；同时，随着任务个数的增加，该算法使用的处理机数和调度长度更显著地少于其它算法，因此，本算法具有更高的调度效率。

结束语　高效的任务调度是 HCS 获得高性能的重要因素。目前对于同构环境的任务调度已经有深入研究；对于异构处理环境下，有效的任务调度算法的研究正成为研究热点。对于 Out-Tree 任务图这一并行处理的重要结构，本文结合同构环境下列表调度和任务复制调度的策略，提出一个基于异构环境的 Out-Tree 任务图的贪心调度算法 LTDGS_OT。本算法结合了列表调度和任务复制的调度策略，在每一调度步，均仔细考察各处理机的负载情况，注重充分利用处理机的处理能力的差异并兼顾处理机的负载平衡；同时采用任务复制的策略消除通信开销，力图节省处理机。实验结果表明，相比相关算法，本算法生成的调度具有调度长度较短、使用的处理机较少的特点，其总体性能相比其他算法有显著的改善。

进一步工作将集中于如何平衡衡量调度质量的各指标，比如平衡平均 *makespan* 值和使用的处理机数目等；这一扩充性研究结果可能会包括在可用处理机数目可能不充足的情形下给出 *makespan* 减少的某些界等。另外，如何更高效地使用任务复制技术，从而进一步缩短调度长度，也是值得研究的问题。

参 考 文 献

[1] Lotfifar F, Shahhoseini H S. A Low-Complexity Task Scheduling Algorithm for Heterogeneous Computing Systems[C]// Third Asia International Conference on Modelling & Simulation. 2009(5):596-601

[2] Hagras T, Janecek J. An approach to compile-time task scheduling in heterogeneous computing systems[C]// Proc. International Conf. on Parallel Processing Workshops. 2004:182-189

（下转第 146 页）

广义签密在一般签密方案能同时提供机密性和认证性的基础上，可以只实现机密性或认证性一种功能，在节约时间和成本的基础上满足了不同的需求环境，因而具有更广泛的应用前景。

参 考 文 献

[1] Shamir A. Identity-based cryptosystems and signature schemes [C]// Proceeding of Crypto'84, LNCS196. Berlin: Springer-Verlag, 1984: 47-53

[2] Boneh D, Frankklin M. Identity Based Encryption From the Weil Pairing [C] // Proceeding of Crypto' 01, LNCS2139. Berlin: Springer-Verlag, 2001: 213-229

[3] Al-Riyami S S, Paterson K G. Certificateless public key cryptography [C]// Proceeding of ASIACRYPT 2003, LNCS2894. Berlin: Springer-Verlag, 2003: 452-473

[4] Girault M. Self-certified public keys[C]// Proceeding of Eurocrypt '91, LNCS547. Berlin: Springer-Verlag, 1991: 490-497

[5] 杜红珍. 数字签名技术的若干问题研究[D]. 北京：北京邮电大学，2009

[6] Chen Xiao-feng, Zhang Fang-guo, Kim K. A New ID-based Group Signature Scheme from Bilinear Pairings [C] // Proceedings of WISA'03. 2003: 585-592

[7] Liao Jian, Xiao Jun-fang, Qi Ying-hao, et al. ID-based signature scheme without trusted PKG [C] // Proceeding of CISC 2005, LNCS3822. Berlin: Springer-Verlag, 2005: 53-62

[8] 周亮，李大鹏，杨义先. 基于身份的无需可信任 PKG 的签名方案[J]. 通信学报，2008，29(6)：8-12

[9] Liu Jing-wei, Sun Rong, Kou Wei-dong, et al. Efficient ID-based Signature Without Trusted PKG[EB/OL]. http://eprint.iacr.org /2007/135, 2007-04-17

[10] 徐玲玲. 基于身份的无可信 PKG 的签名体制[D]. 济南：山东大学，2008

[11] Zheng Yu-liang. Digital signcryption or how to achieve cost (signature & encryption)<<cost (signature)+cost (encryption) [C] // Proceeding of Crypto' 97, LNCS1294. Berlin: Springer-Verlag, 1997: 165-179

[12] 韩益亮，杨晓元. ECDSA 可公开验证广义签密[J]. 计算机学报，2006，29(11)：2003-2012

[13] Han Yi-liang. Generalization of signcryption for resources-constrained environments[J]. Wireless Communication and Mobile Computing, 2007, 7(7): 919-931

[14] Han Yi-liang, Gui Xiao-lin. Adaptive secure multicast in wireless networks[J]. International Journal of Communication Systems, 2009, 22(9): 1213-1239

[15] 魏靓，张串绒，郑连清. 基于广义签密的移动 Ad hoc 网络密钥管理方案[J]. 计算机工程与应用，2010，46(32)：6-9

[16] 杨晓元，李秀广，郝斌，等. 基于广义签密和 DWT 技术的图像水印算法[J]. 计算机工程与应用，2007，43(36)：86-88

[17] Yang Xiao-yuan, Li Mao-tang, Wei Li-xian, et al. New ECDSA-Verifiable Multi-Receiver Generalization Signcryption [C]// The 10th IEEE International Conference on High Performance Computing and Communications. 2008: 1042-1047

[18] Han Yi-liang, Gui Xiao-lin. BPGSC: Bilinear pairing based generalized signcryption scheme[C]// 2009 Eighth International Conference on Grid and Cooperative Computing. 2009: 76-82

[19] 杨晓元，黎茂堂，魏立线. ECDSA 可公开验证广播签密[J]. 解放军理工大学学报：自然科学版，2009，10(4)：324-328

[20] Zhang Chuan-rong, Zhang Yu-qing. Secure and Efficient Generalized Signcryption Scheme Based on a Short ECDSA[C]// Proc of IIH-MSP'2010. 2010: 466-469

[21] 冀会芳，韩文报，刘连东. 标准模型下多个 PKG 的基于身份广义签密[J]. 电子与信息学报，2011，33(5)：1204-1210

[22] 冀会芳，韩文报，刘连东. 高效的无证书广义签密方案[J]. 四川大学学报：工程科学版，2011，43(5)：133-139

[23] Lai S, Kushwah P. ID-based generalized signcryption[EB/OL]. http://eprint.iacr.org/2008/084, 2008-02-26

[24] Yu Gang, Ma Xiao-xiao, Shen Yong, et al. Provable secure identity based generalized signcryption scheme[J]. Theoretical Computer Science, 2010, 411(40-42): 3614 -3624

[25] Kushwah P, Lai S. An efficient identity based generalized signcryption scheme[J]. Theoretical Computer Science, 2011, 412 (45): 6382-6389

[26] 赵静. 高效的基于身份和对映射的广义签密方案的研究[D]. 秦皇岛：燕山大学，2010

[27] 刘景伟，孙蓉，马文平. 高效的基于 ID 的无证书签名方案[J]. 通信学报，2008，29(2)：87- 94

[28] Barbosa M, Farshim P. Certificateless signcryption[C]// ACM Symposium on Information, Computer and Communications Security-ASIACCS 2008. ACM, 2008: 369-372

[29] Selvi S S D, Vivek S S, Rangan C P. Cryptanalysis of certificateless signcryption schemes and an efficient construction without pairing[EB/OL]. http://eprint.iacr.org/2009/298, 2009-06-22

(上接第 110 页)

[3] Topcuoglu H, Hariri S, Wu M Y. Performance-effective and low-complexity task scheduling for heterogeneous computing[J]. IEEE Trans. Parallel and Distributed Systems, 2002, 13(3): 260-274

[4] Kwok Y-K, Ahmad I. Dynamic Critical-Path Scheduling: An Effective Technique for Allocating Task Graphs to Multiprocessors[J]. Parallel and Distributed Systems, 1996(7): 506-521

[5] Darbha S, Agrawal D P. Optimal Scheduling Algorithm for Distributed-Memory Machines[J]. IEEE Trans. Parallel and Distributed Systems, 1998, 9(1): 87-95

[6] Liu Zhen-ying, Fang Bin-xing, Zhang Yi. TSA-OT, An Algorithm Scheduling An Out-Tree DAG[J]. Chinese Journal of Computers, 2001, 24(4): 1-5

[7] 张建军，李庆华，瞿勇. 基于任务复制的调度算法[J]. 计算机工程与设计，2009，30(8)：1896-1899

[8] Omara F A, Arafa M M. Genetic Algorithms for Task Scheduling Problem[J]. Journal of Parallel and Distributed Computing, 2010, 70(1): 13-22

[9] Zhong Yi-wen, Yang Jian-gang, Qia Heng-nian. Hybrid Genetic Algorithm for Tasks Scheduling in Heterogeneous Computing Systems[C]// Proceedings of the Third International Conference on Machine Learning and Cybernetics. Shanghai. 2004(8): 26-29

[10] Darbha S, Agrawal D P. A task duplication based scalable scheduling algorithm for distributed memory systems[J]. Journal of parallel and Distributed Computing, 1997, 46(1): 15-27

[11] Yang Jia-dong, Xu Hua, Jia Pei-fa. Task Scheduling for Heterogeneous Computing based on Bayesian Optimization Algorithm [C]// 2009 International Conference on Computational Intelligence and Security. 2009: 112-117