优化能耗的可变电压禁忌任务调度算法

康 雁
(云南大学软件学院 昆明 650091)

摘 要 能耗是影响异构式并行和分布式系统性能的一个重要因素,动态电压缩放(DVS)技术通过将处理器降低到不同频率来达到有效地节约能耗的目标。通常DVS技术包含任务调度及空闲时间片分配两阶段。当前绝大部分研究均针对时间片分配阶段,而在此考虑的是任务分配与空闲时间片间的关系。为了降低异构分布式系统的能耗,提出了一个利用禁忌(Tabu)策略进行调度的DVS算法。此算法首先调度用有向无环图(DAG)表示的任务集到处理器上,再应用禁忌策略来改进它,通过禁止任务再调度到特定处理器,从而增加时间片,分配阶段可用的空闲时间片达到进一步减少能耗的目标。仿真结果表明,本算法能有效地减少计算机系统的能耗。

关键词 调度算法,动态电压缩放,异构式系统,能耗最小化,禁忌搜索
中图法分类号 TP302 **文献标识码** A

Variable Voltage Tabu Task Scheduling Algorithm for Optimizing Energy Consumption

KANG Yan
(School of Software, Yunnan University, Kunming 650091, China)

Abstract Energy consumption is a critical issue in heterogeneous parallel and distributed systems. Dynamic voltage scaling(DVS) is a powerful technique to achieve energy saving by slowing down the processor into multiple frequency levels, and DVS algorithms typically consist of the assignment of tasks and the allocation of the slack. While most research focuses on the allocation phase, we considered the relation between the slack time and the assignmented of the tasks, and presented a Tabu-based Scheduling algorithm for energy minimization on heterogeneous distributed system. The algorithm finds an initial schedule for the tasks represented by directed acyclic graph(DAG), and improves it by using a Tabu search strategy. The total energy consumption of the system is reduced further by using the Tabu strategy to forbid the task to assign onto specific processors and create more idle time slices for the slack allocation phase. Simulation results indicate that our algorithm achieves substantial energy savings on computer systems.

Keywords Scheduling algorithm, Dynamic voltage scaling, Heterogeneous system, Energy minimization, Tabu search

近年来,随着计算机的广泛使用和处理器性能的不断提高,如何节约计算机能耗成为一个至关重要的问题。系统能耗还与其他很多问题密切相关,例如便携式设备的能耗成了性能增加的限制因素;能耗太大导致散热问题严重,影响正常工作。近年来,系统芯片(SOC)技术已经非常流行,集成度很高,散热成为系统设计者必须解决的难题,而低能耗设计是解决散热问题的最有效方法。

降低电子系统的能耗可用硬件节能和软件节能两种方法实现。

硬件节能技术是指从系统构造、工作原理等方面入手,降低系统能耗,如选用低能耗器件、采用异步电路体系设计等。但光从硬件着手还是不能最终解决问题。尽管低能耗设计技术越来越先进,但由于不断提高的处理器性能,再加上越来越复杂的存储设备、通信部件和越来越复杂的用户接口,使处理器的能耗仍不断提高。

软件节能技术通过改变系统的运行行为来达到降低系统能耗的目的。如在系统工作过程中,根据运行状况将器件从工作状态转入睡眠状态。但在硬件设备被唤醒的过程中,能量又会有很大损耗,同时系统性能也会受很大影响。

动态电压缩放(DVS)技术则是软件节能技术的另一种形式,它是一种新型、有效的低能耗设计方法。由于CMOS电路的能耗和供电电压的平方成正比关系,频率也成正比,因此降低供电电压和频率可大幅度降低能耗。DVS技术通过动态地缩放处理器的工作电压和频率,在给定的时间范围内完成任务并降低处理器的能耗[1-3]。

动态电压缩放就是在一定的范围内,通过动态地给定处理器的工作电压和相应的时钟频率,从而达到减少能耗的目的。对于处理器来说,电压的降低是减少处理器能耗的主要途径。然而降低电压的同时也必须随之降低处理器的频率,但这又将带来处理器的延迟,因此处理器的电压和频率不能简单地被降低,必须在规定的时间范围内根据各任务在处理器上的运行情况来减少能耗。任务在各处理器上的运行时间

到稿日期:2009-11-20 返修日期:2010-01-20 本文受国家自然科学基金项目(10561009),国家自然科学基金项目(60763008),云南省科学基金(2005F0010Q)资助。

康 雁(1972—),女,博士,副教授,主要研究方向为计算复杂性问题、超启发算法,E-mail:kangyanpu@gmail.com。

及能耗给定的前提下，本文对任务在处理器上的情况进行调整，通过禁忌策略改变任务在处理器上的调度，从而增加了任务可利用空闲时间片，进一步地减少了能耗。最后用大量的仿真实验验证了此算法的有效性。

1 问题模型

1.1 能耗模型

处理器的能耗为

$$P_d = C_{ef} \cdot V_{dd}^2 \cdot S \tag{1}$$

式中，V_{dd} 是供电电压；C_{ef} 是有效转换电容；S 是处理器的时钟频率，计算公式为

$$S = k \cdot (V_{dd} - V_t)^2 / V_{dd} \tag{2}$$

式中，V_t 是门限电压，k 是常数，它表示处理器的速度和供应电压间基本上是线性关系。c/S 即为任务执行所需的时间 t，其中 c 为执行任务所需的循环数。

任务在处理器上的能耗为

$$E = P_d \cdot t = C_{ef} \cdot V_{dd}^2 \cdot c \tag{3}$$

DVS 技术通过降低电压 V_{dd} 即延长任务执行时间 t，能够有效地减少任务能耗 E。针对单处理器和同构多处理器上的独立任务集的 DVS 技术已有大量研究。为了有效节约大量如天气预报、图像处理、数字信号传输、实时、分布式数据库等计算任务的能耗需求，本文针对异构式系统上有不同运行时间、不同能耗需求并且相互间有先序关系的任务集进行了研究。

1.2 任务模型

本文研究的任务问题模型可简述为一个有向无环图 $G=(T,P,C,D,E)$，$T=\{T_i \mid i=1,2,\cdots,n\}$ 是图中的顶点集，表示由 n 个任务组成的任务集。任务间的先序关系表示任务间的依赖关系，亦即前一项任务完成后后一项任务才能开始。无任何先序任务的任务称为初始任务，而无任何后序任务的任务为终止任务。不失一般性，可设任务集中有且仅有一个初始任务 T_0、一个终止任务 T_n。若有多个开始任务和多个终止任务，可使它们与计算费用为零的一个虚拟初始结点和一个虚拟终止结点以通讯费用为零的边相连即可；$P=\{P_j \mid j=1,2,\cdots,m\}$ 为处理器集，各处理器可以是异构处理器；$C=\{C_{i,j} \mid i=1,2,\cdots,n, j=1,2,\cdots,m\}$ 为计算时间集，$C_{i,j}$ 表示任务 V_i 在处理器 P_j 上的计算时间；$D=\{D_{i,j} \mid i,j=1,2,\cdots,n\}$ 为结点间的传输边集，它的值表示有先序关系的任务 V_i 和 V_j 间的传输时间。当有先序关系的两个任务在同一处理器上运行时，它们间传输时间为零；$E=\{E_{i,j} \mid i=1,2,\cdots,n, j=1,2,\cdots,m\}$ 为任务能耗集，$E_{i,j}$ 表示任务 V_i 在处理器 P_j 上的处理能耗。为了简化模型，我们忽略数据传输所需的能耗。

2 迭代式禁忌调度算法

2.1 HEFT 策略

DVS 技术根据各任务在处理器最高电压下的运行时间进行调度。在其完成时间不超过最后完成期限的情况下，利用其最早开始时间和最迟开始时间差延长运行时间，从而减少能耗。

在满足任务先序关系的前提下，使任务在各处理器上的总完成时间最早的任务调度问题，已证明是 NP-完全问题，亦即最优解只能在大量的搜索后才能得到，因此目前研究的主要是在尽可能短的时间内找到近似最优解的启发式调度算法[8-10]。为了有效地把任务调度到处理器上，我们利用了已有的 Heterogeneous Earliest-Finish-Time(HEFT)启发式任务调度策略[11]。

HEFT 策略分为排序及分配两个阶段。第一阶段排序是将任务按计算得到的优先级非递增排序。每个任务的优先级 W_i 按下列公式计算：

$$W_i = M_i + \min_{T_j \in Succ(T_i)} \{D_{i,j} + W_j\} \tag{4}$$

式中，M_i 为任务 T_i 在所有处理器上的平均处理时间；$Succ(T_i)$ 是 T_i 的后继任务集。显然，终止任务 T_n 的优先权 $W_n = M_n$，W_i 是按任务执行顺序从后往前地计算得到。

HEFT 策略的第二阶段分配是按优先级从高到低依次取出各任务 T_i，在满足先序关系的前提下分配到使目标函数 $EFT_{i,j}$ 最小的处理器 P_j 上。目标函数 $EFT_{i,j}$ 是任务 T_i 可以在处理器 P_j 上完成的最早时间，其计算公式如下：

$$EFT_{i,j} = C_{i,j} + \max_{T_k \in Pred(T_i)} \{A_i, (EFT_k + D_{k,i})\} \tag{5}$$

式中，A_j 是任务 T_i 在处理器 P_j 上可以开始的最早时间，它通过计算调度在处理器 P_j 上任务的后继空闲时间得到。其中，$Pred(T_i)$ 是任务 T_i 的先序任务集，EFT_k 为任务 T_i 的先序任务 T_k 的最早完成时间，调度完成后，系统总能耗为

$$E = \sum_{i=1}^{n} P_{i\max} \cdot t_{i\min} \tag{6}$$

式中，$P_{i\max}$ 为处理器的最高能耗，$t_{i\min}$ 为任务在处理器上的最短运行时间。

2.2 ISA 策略

当处理器上调度的任务间存在可用空闲时间片时，使用 DVS 技术后得到的系统能耗减少量为

$$\Delta E = \sum_{i=1}^{n} E_i(t_i) - E_i(t_i + \Delta t_i) \tag{7}$$

式中，$t_i + \Delta t_i$ 是使用 DVS 技术后任务 T_i 的运行时间，Δt_i 将等于或小于任务可使用的最大空闲时间片 $Slack_i$，$Slack_i$ 由下式计算得到

$$Slack_i = LFT_i - EFT_i \tag{8}$$

式中，LFT_i 是任务 T_i 的最迟完成时间，它按下述公式计算得到

$$LFT_i = \max_{T_k \in Succ(T_i)} \{LFT_k - C_{k,l} - D_{i,k}\} \tag{9}$$

LFT_i 是按任务执行顺序从后往前地计算得到，终止任务的最迟完成时间为任务集的时间期限。

任务 T_i 延长运行时间后，得到的新能耗为

$$E_i(t + \Delta t) = E_i(t) \cdot \frac{V_{idd}^2}{V_{i\max}^2} \tag{10}$$

式中，V_{idd} 是任务 T_i 新的运行电压，计算公式为

$$V_{idd} = V_{it} + V_{i0}/(2 * d_i) + \sqrt{(V_{it} + V_{i0}/(2 * d_i))^2 - V_{it}^2} \tag{11}$$

式中，$d_i = (t + \Delta t)/t$ 为前后运行时间比。V_{i0} 在给定的处理器上通常为一个常数，由该处理器的最高电压 $V_{i\max}$ 和门限电压 V_{it} 决定，它按如下公式计算得到

$$V_{i0} = (V_{i\max} - V_{it})^2 / V_{i\max} \tag{12}$$

对于异构多处理器上有先序关系的任务集，空闲时间片往往由多个任务所共享，将时间片分配给单一的任务不能达到最优。由于系统的异构性，将时间片有效地分配到各个任务间很难减少能耗。为此，文献[12]中提出 Iterative Slack

Allocation(ISA)策略。它是一个迭代式空闲时间片分配策略，当异构式系统中有多任务共享空闲时间片时，迭代地将细分后的时间片分配给当前能耗量下降最大的任务，并修改相应的任务运行时间和电压。重复上述步骤，直至无空闲时间片为止。ISA策略可以有效地在异构式处理器的任务集间分配时间片降低能耗。

2.3 可变电压禁忌任务调度算法

因为DVS技术涉及任务在处理器上的分配、任务在处理器上的执行顺序以及任务在不同处理器上的运行电压及运行时间，所以单一地以运行时间或能耗为目标进行调度并不能达到能耗最小。在此，我们提出了可变电压禁忌任务调度策略(Variable voltage tabu task scheduling，VVTTS)算法，它通过调整已有调度增加任务的空闲时间，有效地减少了能耗。

在此，我们针对任务的调度采用了禁忌策略。当不具备先序关系的任务分配到同一处理器上时，会增加任务间的先序关系，影响任务的最早开始时间和最迟开始时间，从而减少空闲时间。若任务 T_i 的空闲时间因为后继任务的增加而减少，则利用禁忌策略调整已有任务的调度。

禁忌搜索于1986年由Glover[13]提出并形式化它是局部领域搜索算法的推广，是人工智能在组合优化算法中的一个成功应用。禁忌策略通过禁止重复前面的工作回避了局部领域搜索陷入局部极小值陷阱。禁忌搜索算法用一个禁忌表记录到达过的局部极小值点，在下一次搜索中，利用禁忌表中的信息不再或有选择地搜索这些点，以此来跳出局部极小值点。

我们使用了禁忌表中的两个主要指标：禁忌对象和禁忌长度。我们采用的禁忌对象是任务在处理器上的调度，禁止任务调度在同一处理器上。禁忌长度为3，即3次禁忌任务的调整后能耗没有减少，就停止任务的调整。

VVTTS算法可描述为：

步骤1

步骤1.1 按HEFT策略得到任务的优先序列；

步骤1.2 在满足先序关系的前提下按优先序列将任务分配到使其完成时间最早的处理器上。

步骤2

步骤2.1 根据此调度中任务可利用的空闲时间，按优先顺序从前往后地寻找满足禁忌规则的任务 T_i；

步骤2.2 将禁忌任务 T_i 调整到使其完成时间最早的非禁忌处理器上；

步骤2.3 按优先序列将 T_i 后的任务分配到使其完成时间最早的处理器上；

步骤2.4 利用DVS技术计算得到系统能耗；

步骤2.5 若能耗不小于调整前，取消对此任务的禁忌。

步骤3 重复步骤2操作，直至无满足禁忌规则的任务或连续3次调整后能耗不能减少。

3 实验和分析

本文使用JAVA语言对各算法进行了编程，并在有1G内存的Intel Xeon处理器上进行了仿真实验。为了分析算法的性能，在此采用两组实验数据：一是给定的具体实例；一是随机生成的一组实验数据，生成20到200个任务，分配到10到180个处理器上，任务间的关系是随机生成的，同一处理器上任务的能耗相差最多为3倍。为了比较分析算法，我们还编程实现了另外两个算法：HISA算法是使用HEFT和ISA策略得到；EISA算法是使用势能为目标函数的调度策略和ISA策略得到。首先将图1中任务集分配到两个处理器进行仿真计算，表1给出图1中各任务的计算时间、传输时间及能耗。表1中 $C_{i,j}$ 和 $E_{i,j}$ 为任务 T_i 在处理器 P_j 上的运行时间和能耗，而 D_i 为任务 T_i 的先序任务与其不在同一处理器时传送数据的时间。当时间期限 $deadline=1.57$ 时，表2、表3和表4分别为HISA，EISA和VVTTS 3个算法得到的任务 T 调度到处理器 P 上的运行起止时间 CP_1 和能耗 E_1 以及利用DVS后的运行起止时间 CP_2 和能耗 E_2。表5是以 $deadline+=deadline*10\%$ 重复10次增加时对应的HISA，EISA和VVTTS 3个算法的能耗 E_{HISA}，E_{EISA} 和 E_{VVTTS}，$R1\%=100*(E_{EISA}-E_{HISA})/E_{HISA}$ 和 $R3\%=100*(E_{VVTTS}-E_{HISA})/E_{HISA}$ 分别为EISA、VVTTS算法与HISA算法相比能耗减少的百分比，$R2\%=100*(E_i-E_0)/E_0$ 为第 i 次增加 $deadline$ 时HISA算法能耗 E_i 与初始能耗 E_0 相比减少的百分比。从表2到表4可以看出，以目标函数能耗最小进行调度并使用DVS后得到的能耗并不接近最优解。而随着最后完成期限的延长，各算法得到的能耗量减少量在增加，但减少比例逐渐减少，同时各算法间的能耗减少比例基本不变。

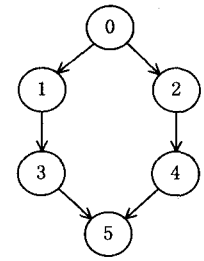

图1 任务的DAG图

表1 任务的计算时间、传输时间及能耗

	T_0	T_1	T_2	T_3	T_4	T_5
$C_{i,1}$	0.50	0.40	0.30	0.35	0.10	0.20
$C_{i,2}$	0.70	0.25	0.45	0.40	0.50	0.20
$E_{i,1}$	58	46	32	38	15	23
$E_{i,2}$	73	29	48	53	52	23
D_i	0	0.03	0.03	0.02	0.02	0.02

表2 EISA算法的计算结果

P	V	CP_1	CP_2	E_1	E_2
1	0	0.00—0.50	0.00—0.50	58	58.00
1	3	0.80—1.15	0.80—1.25	38	22.99
1	4	1.25—1.35	1.25—1.35	15	15.00
2	1	0.53—0.78	0.53—0.78	29	29.00
2	2	0.78—1.23	0.78—1.23	48	48.00
2	5	1.37—1.57	1.37—1.57	23	23.00

表3 HISA算法的计算结果

P	V	CP_1	CP_2	E_1	E_2
1	0	0.00—0.50	0.00—0.54	58	49.29
1	2	0.50—0.80	0.54—0.86	32	28.05
1	3	0.80—1.15	0.86—1.24	38	33.64
1	4	1.15—1.25	1.24—1.35	15	10.62
1	5	1.25—1.45	1.35—1.57	23	19.65
2	1	0.53—0.78	0.57—0.84	29	24.79

表4 VVTTS算法的计算结果

P	V	CP_1	CP_2	E_1	E_2
1	0	0.00—0.50	0.00—0.56	58	45.78
1	2	0.50—0.80	0.56—1.12	32	9.37
1	4	0.80—0.90	1.12—1.33	15	3.48
2	1	0.53—0.78	0.59—0.87	29	22.89
2	3	0.78—1.18	0.87—1.35	53	38.28
2	5	1.18—1.38	1.35—1.57	23	18.15

表5 各算法计算结果比较

E_{EISA}	E_{HISA}	E_{VVTTS}	Deadline	R1%	R2%	R3%
195.99	166.04	137.95	1.57	−18.04	0	16.92
161.27	137.05	113.62	1.73	−17.67	17.46	17.10
134.59	114.57	95.21	1.88	−17.47	31.00	16.90
114.02	97.33	80.93	2.04	−17.15	41.38	16.85
97.76	83.71	69.64	2.20	−16.78	49.58	16.81
84.80	72.77	60.57	2.36	−16.53	56.17	16.77
74.26	63.84	53.14	2.51	−16.32	61.55	16.76
65.53	56.42	47.02	2.67	−16.15	66.02	16.66
58.29	50.26	41.89	2.83	−15.98	69.73	16.65
52.18	45.05	37.55	2.98	−15.83	72.87	16.65
46.98	40.62	33.86	3.14	−15.66	75.54	16.64

为了进一步分析算法的有效性，我们对任务数 V_{num} 从 20 到 200、处理器数 P_{num} 从 10 到 180 进行了随机实例计算，并计算得到相应的能耗。其中 $DE1=100*(E_{HISA}-E_{VVTTS})/E_{HISA}$ 为 HISA 算法与 VVTTS 算法相比得到的能耗百分比变化范围，$DE2=100*(E_{HISA}-E_{EISA})/E_{HISA}$ 为 VVTTS 算法与 HISA 算法的能耗减少百分比变化范围。从表 6 可以看出，EISA 算法得到的系统能耗量较大，并且任务数和处理器数越多时系统总能耗比其它算法越大，最多达到 HISA 算法的 20 倍。而随着任务数和处理器数的增多，任务 VVTTS 算法得到的能耗量约为 HISA 算法的 80%～95%。注意到处理器过少或过多时总的任务调度可行解较少，因此各算法的区别并不大。

表6 随机计算结果比较

V_{num}	P_{num}	DE1	DE2
200−150	180−140	6−20	−2000−100
200−150	140−100	6−20	−1000−50
200−150	100−60	6−20	−400−20
200−150	60−20	6−20	−100−10
150−100	120−90	5−20	−1400−100
150−100	90−60	5−20	−1000−40
150−100	60−30	5−20	−350−1
100−50	80−60	5−20	−1000−100
100−50	60−40	5−20	−400−50
100−50	40−20	5−20	−300−0.5
50−40	40	3−20	−1000−40
50−40	20	3−20	−900−10
50−40	10	3−30	−300−0.5
30−20	10	3−30	−100−1

结束语 动态电压缩放技术可有效地减少计算机系统的能耗。本文研究了如何在分布式和异构式系统中利用动态电压缩放技术减少能耗。针对任务在不同处理器上的运行时间、空闲时间片和能耗的情况，我们提出了利用禁忌策略的迭代式动态电压缩放算法，算法通过禁忌任务分配在某些处理器上获得更多的可利用空闲时间，经过动态电压缩放后减少了任务在处理器上的能耗量，最终用仿真实验验证了算法的有效性。

参考文献

[1] Burd T D, Pering T A, Stratakos A J, et al. A Dynamic Voltage Scaled Microprocessor System[J]. IEEE Journal of Solid-State Circuits, 2000, 35(11): 1571-1580

[2] Manzak A, Chakrabarti C. Variable voltage task scheduling algorithms for minimizing energy[J]. IEEE Journal of Transactions VLSI System, 2003, 11(2): 270-276

[3] Yi Hui-Zhan, Yang Xue-jun. Toward the Optimal Configuration of Dynamic Voltage Scaling Points in Real-time Applications [J]. Journal of Computer Science and Technology, 2006, 21(6): 893-900

[4] Chowdhury P, Chakrabarti C. Static task-scheduling algorithms for battery-powered DVS systems[J]. IEEE Journal of Transactions VLSI System, 2005, 13(2): 226-237

[5] Aydin H, Melhem R, Mossé D, et al. Power-Aware Scheduling for Periodic Real-time Tasks[J]. IEEE Journal of Transactions on Computers, 2004, 53(5): 584-600

[6] Wong I Y, Casey R G, Wahl E M. Document Analysis System [J]. IBM Journal of Research Develop, 1982, 26(6): 647-656

[7] Jejurikar R, Gupta R. Optimized Slowdown in Real-Time Task Systems[J]. IEEE Journal of Transactions on Computers, 2006, 55(12): 1588-1598

[8] Braun T D, Siegel H J, Beck N, et al. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems[J]. Journal of Parallel and Distributed Computing, 2001, 61(6): 810-837

[9] Zhou Lan, Sun Shixin. Scheduling algorithm based on critical tasks in heterogeneous environments[J]. Journal of Systems Engineering and Electronics, 2008, 19(2): 398-405

[10] Bajaj R, Agrawal D P. Improving scheduling of tasks in a heterogeneous environments[J]. IEEE Journal of Transactions on Parallel and Distributed Systems, 2004, 15(2): 107-118

[11] Topcuoglu H, Hariri S, Wu M. Performance-effective and Low-complexity Task Scheduling for Heterogeneous Computing[J]. IEEE Journal of Transactions on Parallel and Distributed Systems, 2002, 13(3): 260-274

[12] Schmitz M T, Al-Hashimi B M, Eles P. Iterative Schedule Optimization for Voltage Scalable Distributed Embedded System[J]. ACM Journal of Transactions on Embedded Computing Systems, 2004, 3(1): 182-217

[13] Glover F. Future Paths for Integer Programming and Links to Artificial Intelligence[J]. Journal of Computers and Operations Research, 1986, 13(5): 533-549

[14] Thesen A. Design and Evaluation of Tabu Search Algorithms for Multiprocessor Scheduling[J]. Journal of Heuristics, 1998, 4(2): 141-160

(上接第 256 页)

[7] 彭可，陈燕红，唐宜清. 一种室内环境运动目标检测混合算法[J]. 计算机工程与应用，2008，44(5)：239-241

[8] 代科学，李国辉，涂丹，等. 监控视频运动目标检测减背景技术的研究现状和展望[J]. 中国图象图形学报，2006，11(7)：919-917

[9] Friedman N, Russell S. Image segmentation in video sequences: A probabilistic approachf[A]//Proceedings of the 13rd Conference on Uncertainty in Artificial Intelligence[C]. Rhode Island, USA, 1997: 175-181

[10] 周凯. 交通视频监控系统中运动目标特征提取与数据传输实时性保证的研究[D]. 上海：同济大学，2007

[11] Doshi A, Trivedi M. "Hybrid Cone-Cylinder" Codebook Model for Foreground Detection with Shadow and Highlight Suppression[C]//Proceeding of IEEE International Conference on Video and Signal Based Surveillance. Nov. 2006: 19-19

[12] 王华伟，李翠华，施化. 基于 HSV 空间和一阶梯度的阴影剪除算法[J]. 计算机工程与应用，2005，41(8)：43-44