Optimal Cyclic Scheduling on Parallel Processors with Special Precedence Constrains

Natalia Grigoreva

[1] St.Petersburg State University, St.Petersburg, Russia,

Conference OPTIMA2023

Outline

In classical scheduling, a set of jobs V is executed once, and the goal is to generate an optimal schedule. The usual objective function is the completion time of the scheduled tasks also referred to as makespan and the goal is to minimize the makespan.
A cyclic scheduling problem is a scheduling problem in which some set of tasks V is to be repeated an infinitely number of times. These approaches are also applicable if the number of loop repetitions is large enough.

Cyclic scheduling has multiple applications, such as robotics , manufacturing systems, communications and transport or multiprocessor computing.

Cyclic scheduling applications usually deal with a periodic schedule, which is a schedule of one iteration that is repeated within a fixed time interval called the period (or cycle time).

The aim of cyclic scheduling is to find a periodic schedule with the minimum period.

Cyclic scheduling is not less difficult than non-cyclic scheduling since any non-cyclic scheduling problem polynomially reduces to a cyclic problem where successive iterations must not overlap.

Four cyclic versions of scheduling problems

We propose the algorithm for the cyclic version of the problem
$P|prec, p_j = 1|C_{\max}$,
$P|pmtp, prec|C_{\max}$,
the problem with independent jobs $P||C_{\max}$,
and the cyclic version of the problem $P|prec|C_{\max}$.

Cyclic Scheduling Problem

Let $V = \{v_1, ..., v_n\}$ be a set of generic operations.
$< v_i; k >$ is the k-th occurrence of the generic operation v_i.
Precedence relations are defined by a graph $G = (V, E)$.
Each arc $(v_i, v_j) \in E$ is supplied by two values $L_{ij} = p(v_i)$ processing time and H_{ij}.
If $(v_i, v_j) \in E$
the task $< v_i; k >$ must be completed before the task $< v_j, k + H_{ij} >$ starts.
Let m identical processors are available to execute the tasks.
As usual, each task $< v_i; k >$ is performed by one processor and, at any instant, one processor may perform at most one task.
A schedule for the set V is the mapping of each task $v_i \in V$ a start time $t(v_i, k)$ and a processor $f(v_i)$.

Cyclic Scheduling Problem

The graph $G = (V, E)$. leads to the following uniform precedence constraints

$$t(v_i; k) + p(v_i) \leq t(v_j; k + H_{ij}).$$

We also postulate that the $k + 1$-th occurrence of operation v_i can only start if the k-th occurrence is finished. Thus, we get the following constraint

$$t(v_i; k) + p(v_i) \leq t(v_i; k + 1).$$

In the following we assume that the constraints are included in graph G by adding loops (i, i) with $L_{ii} = p(v_i)$ and $H_{ii} = 1$ to E.

Cyclic Scheduling Problem

Definition

A schedule is called periodic with cycle time w, if $t(v_i;k) = t(v_i;1) + (k-1)w$ for all $v_i \in V$, $k \geq 1 \in N$.

We can take the length of the schedule $C_{\max}$ as the cycle time. But if we want to minimize the cycle time, then it is possible to increase the length of the schedule, i.e. time from the start of the first task to the end time of the last task
Let's set the cycle time equal to the lower bound and minimize the schedule length.

Lower bound for the optimal cycle time

Consider a circuit μ in graph G. Let $L(\mu) = \sum_{(i,j)\in\mu} p(v_i)$ and $H(\mu) = \sum_{(i,j)\in\mu} H_{ij}$.
Then $z(\mu) := \frac{L(\mu)}{H(\mu)}$ is called the value of μ. The circuits with the maximum value and positive height are called critical circuits. Then the value of a critical circuit $z(G)$ is a lower bound for the optimal cycle time and $LB = \max\{\sum p(v_i)/m, z(G)\}$ is a lower bound on the cycle time.

Cyclic Scheduling Problem

$C_{\max} = \max\{t(v_i;1) + p(v_i) \| \ i \in 1:n\} - \min\{t(v_i;1) \| \ i \in 1:n\}$
This special LPSIP with can be written as:

$$\min C_{\max}$$

$$z = \max\{\sum p(v_i)/m, z(G)\}$$

$$t(v_i;k) = t(v_i;1) + (k-1)z, \forall v_i \in V,$$

$$t(v_i;k) + p(v_i) \leq t(v_i;k+1), \forall v_i \in V, \forall k \geq 1 \in N.$$

$$t(v_i;k) + p(v_i) \leq t(v_j;k), \forall (v_i, v_j) \in E.$$

Periodic scheduling problem with unit processing times. Algorithm 1

Step 1. Define lower bound $LB = \lceil n/m \rceil$ for the optimal cycle time z_{opt}.
Step 2. Define the occurrence vector $\alpha(v_i) := 0$;
Step 3. Find schedule S_L, start times $t(v_i)$ and makespan $C_{\max}$, use procedure ListCG $(G; S_L, C_{\max})$.
Step 4. If $C_{\max} = LB$ then $z_{opt} = C_{\max}$ and optimal cyclic schedule is $\sigma = (t, z_{opt})$, goto step 11 else set $z = LB, k := 0$
Step 5. $k := k + 1$. Define two sets of jobs
$D_k(z) = \{v_i \mid t(v_i) < z\}$ and $F_k(z) = \{v_i \mid t(v_i) \geq z\}$.
Step 6. Set $\alpha(v_i) := \alpha(v_i) + 1$ and $t(v_i) = 0$ for $v_i \in F_k(z)$.
Step 7. Create a new graph: $G_k = (F_k(z), E_k)$,where G_k is a subgraph of $G^* = (V, E)$. $(v_i, v_j) \in E_k$ if and only if $((v_i, v_j) \in E)\&(v_i, v_j \in F_k(z))$.

Step 8. Jobs $F_k(z)$ are ordered in a priority list, by procedure ListCG.
Step 9. At each step the available job with the highest ranking on a priority list is assigned to the free interval in schedule S_L.
Step 10. Define new S_L and $C_{\max}$. If $C_{\max} = LB$ then $z_{opt} = C_{\max}$ and optimal cyclic schedule is $\sigma = (t, z_{opt}, \alpha(v_i))$, goto step 11 else goto step 5.
Step 11. Algorithm generates the feasible schedule $t(v_i) := t(v_i) + z_{opt}\alpha(v_i)$.

Step 1. Generate schedule S_L use algorithm $ListCG(G, S_L, C_{\max})$

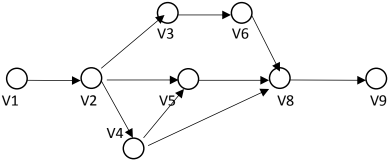


Figure: Task graph $G = (V, E)$, $t(V_i) = 1$;

Example

V1	V2	V3	V5	V7	V8	V9
		V4	V6			

V1	V2	V3	V9	
V5	V7	V4		
V6		V8		

V1	V2	V3
V5	V7	V4
V6	V9	V8

Figure: The schedule S_L, $C_{\max} = 7$; the schedule S_2, $C_{\max} = 4$; the cyclic schedule: $z_{opt} = 3$, $C_{\max} = 8$.

Theorem

Theorem

Algorithm A_1 *generates the feasible cyclic schedule with the minimum cycle time .*

Optimal cyclic scheduling with preemptions

Step1. Define lower bound LB for the optimal cycle time z_{opt}.
$LB = (\sum_{i=1}^{n} p(v_i))/m$.
Step 2. Find schedule S_p and find start times $t_j(v_i)$ and makespan $C_{\max}$, use ListMC $(G; S_p, C_{\max})$.
Step 3. If $C_{\max} = LB$ then schedule S_p is optimal cyclic schedule and cycle time is equal $C_{\max}$, then goto Step 10 else $z := LB, k := 0$.
Step 4. For each job v_i the algorithm defines the start time $t_j(v_i)$ in j-th bloc of execution job v_i, where $j \in 1 : k_i$.
Step 5. $k := k + 1$. Define three sets of jobs:
$D_k(z) = \{v_i \,|(t_{k_i}(v_i) + p_{k_i}(v_i) \le zk)\}$,
$B_k(z) = \{v_i \mid (t_j(v_i) < zk)\&(t_j(v_i) + p_j(v_i) > zk)\}$,
$F_k(z) = \{v_i \mid (t_j(v_i) \ge zk)\&(v_i \notin B_k(z))\}$.
Step 6. The set $D_k(z)$ consist of jobs, which are ended before zk. We interrupt all jobs from $B_k(z)$, which are processing at moment of time zk.
Step 7. For each job v_i from $B_k(z)$ find j_0, such that
$(t_{j_0}(v_i) < zk)\&(t_{j_0}(v_i) + p_{j_0}(v_i) > zk)$

Theorem

Theorem

Algorithm A_2 generate the feasible cyclic schedule with minimum cycle time. This cyclic problem can be solved in $O(n^3)$ time.

Step1. Define lower bound LB for the optimal cycle time z_{opt}.
$LB = \lceil (\sum_{i=1}^{n} p(v_i))/m \rceil$.
Step 2. Define the occurrence vector $\alpha(v_i) := 0$;
Step 3. Find schedule S, start times $t(v_i)$
and makespan $C_{\max}$, use CP $(G; S_L, C_{\max})$.
Step 4. If $C_{\max} = LB$ then schedule S_L is optimal cyclic
schedule and $z_{opt} = C_{\max}$, goto Step 10 else $z := LB$.
Step 5. Define two sets of jobs $D(z) = \{v_i \,| t(v_i) + p(v_i) \leq z\}$
and $F(z) = \{v_i \mid t(v_i) + p(v_i) > z\}$.
Step 6. Set $\alpha(v_i) := \alpha(v_i) + 1$ for $v_i \in F(z)$.

Step 7. Create two new graphs: $G_1 = (D(z), E_1)$ and
$G_2 = (F(z), E_2)$
where G_1 and G_2 are subgraphs of $G = (V, E)$.
Arc $(v_i, v_j) \in E_1$ if and only if $v_i, v_j \in D(z)$ and arc $(v_i, v_j) \in E_2$ if and only if $v_i, v_j \in F(z)$.
Step 8. Construct the new graph $G_{new}(D(z) \cup F(z), E_1 \cup E_2)$.
Step 9. We construct the schedule S_z using algorithm CP by the procedure $CP(G_{new}, S_z, C_{\max})$.
Step 10. $z = C_{\max}$ and the cyclic schedule is $\sigma = (t, z, \alpha(v_i))$.
$t(v_i) := t(v_i) + z\alpha(v_i)$.

Example

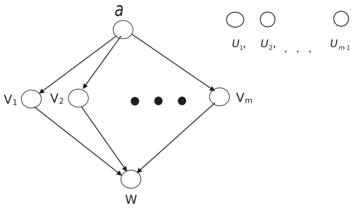


Figure: Task graph $t(a) = \varepsilon$; $t(V_i) = 1$; $t(U_i) = m - 1$; $t(W) = m - 1$

Example

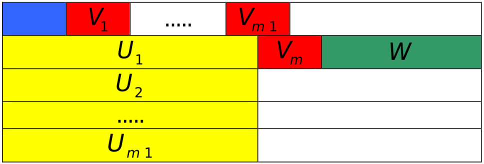
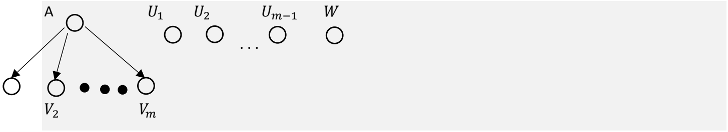
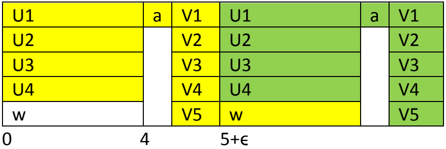


Figure: The schedule 1, makespan $C_{\max} = 2m - 1$

Example



A
U_1
U_2
U_{m-1}
W
. . .
V_2
V_m



U1
a
V1
U1
a
V1
U2
V2
U2
V2
U3
V3
U3
V3
U4
V4
U4
V4
w
V5
w
V5
0
4
5+ϵ

Cyclic scheduling independent jobs on parallel processors

Step1. Define lower bound LB for the optimal cycle time z_{opt}.
$LB = \lceil (\sum_{i=1}^{n} p(v_i))/m \rceil$.
Step 2. Find schedule S_L, start times $t(v_i)$ and makespan $C_{\max}$, use $LPT(V; S_L, C_{\max})$.
Step 3. If $C_{\max} = LB$ then schedule S_L is optimal cyclic schedule and cycle time is equal $C_{\max}$, goto Step 10.
Step 4. Define the sum of processing times of every processors $s_1, s_2, \ldots, s_m$. Lets $\tau(j)$ is the start time of processor j.

Step 5. Renumber processors in non-increasing order:
$s_1 \geq s_2 \geq \ldots, \geq s_m$.
Define $z = \max\{(s_j + s_{m-j+1})/2 \mid j \in 1 : m\}$.
Step 6. Define $f(v_i)$ processor for job v_i.
Step 7. Define the start time of processor j $\tau(1) := 0$,
$\tau(j) = (s_1 - s_j)/2, j \in 2 : m$.
Step 8. If $f(v_i) = j$ then $t(v_i) := t(v_i) + \tau(j)$.
Step 9. Algorithm generates the cyclic schedule $\sigma = (t, z)$.

Example

Consider the worst case example of LPT algorithm with $m = 4, n = 9$

Processing times of jobs $p(V) = (7, 7, 6, 6, 5, 5, 4, 4, 4)$.

LPT algorithm generates the schedule S_L , $C_{\max} = 15$ The optimal schedule has makespan $C_{opt} = 12$.

$t(V) = (0, 0, 0, 0, 6, 6, 7, 7, 11)$ and

$f(V) = (1, 2, 3, 4, 4, 3, 2, 1, 1)$.

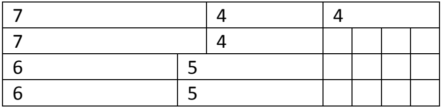


Figure: The schedule S_L, $C_{\max} = 15$

Define the start time of processors
$\tau_1 = 0, \tau_2 = 2, \tau_3 = 2, \tau_4 = 2.$

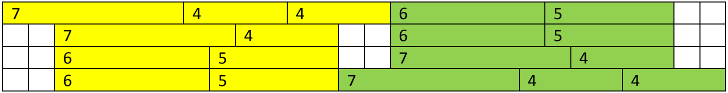


Figure: The cycle schedule S_z, cycle time $z := 13$

Define t(V)=(0,2,2,2,8,8,9,7,11). $z = 13$. The cycle schedule S_z is resource-periodic with period 2: $f(v_i, k) = f(v_i, k+2)$.

Theorem

Theorem

Algorithm A_4 generate the feasible cyclic schedule with $C_{\max}/C_{opt} \leq 4/3 - 1/3m$
This cyclic problem can be solved in $O(nlogn)$ time.

Thanks for attention