The Complexity of two Scheduling Problems of a Chain considering an Energy Constraint

Alix Munier-Kordon

LIP6, Sorbonne Université, Paris

Paris, March 2023
1. Tasks Definition

2. Scheduling the chain on an hybrid platform (GPU/CPU)

3. Scheduling the chain on a chain of AI accelerators

4. Conclusion and perspectives
Tasks model

- $\mathcal{T} = \{1, \ldots, n\}$ a set of non preemptive tasks;
- $1 \rightarrow 2 \rightarrow 3 \ldots \rightarrow n$ a chain of usual precedence constraints: any task $i \in \mathcal{T} - \{0\}$ can start its execution once $i - 1$ is completed.

This simple model allows to consider several concrete situations:

- Streaming applications (JPEG, H264..etc);
- An HPC application alternating computing and I/O phases [Jeannot et al. 2021];
- A Neural Network Architecture.
Platforms consume energy due to the CMOS circuits:
- Dynamic power is associated to the switching activity of the transistors;
- Static power is independent of the activity of the circuit.

When the platform is fixed, we thus consider only Dynamic power.

Models for the evaluation of the total energy consumed exist, and several authors considered these models to consider scheduling problem minimizing the energy [Huang et al. 2012, Pallez and Benoit 2016,...]

However, these works are based on particular models to evaluate and manage the energy consumption. A good model for the energy is not always available. In this case, we have to consider particular values manually measured and depending on tasks or the machine configurations.
Tasks Definition
Scheduling the chain on an hybrid platform (GPU/CPU)
Scheduling the chain on a chain of AI accelerators
Conclusion and perspectives

**Scheduling the chain on an hybrid platform (GPU/CPU)**

The platform considered is composed by two classes of machines: \( \mathcal{A} \) and \( \mathcal{B} \).

- \( p_{i,\mathcal{A}} \) (resp. \( p_{i,\mathcal{B}} \)) is the execution time of task \( i \in \mathcal{T} \) on a machine of type \( \mathcal{A} \) (resp. \( \mathcal{B} \));
- \( e_{i,\mathcal{A}} \) (resp. \( e_{i,\mathcal{B}} \)) is the energy required to execute \( i \in \mathcal{T} \) on a machine of type \( \mathcal{A} \) (resp. \( \mathcal{B} \));

Let \( \mathcal{T}' \subseteq \mathcal{T} \) be the tasks performed by a machine \( \mathcal{B} \);
- The length of the schedule associated to \( \mathcal{T}' \) equals
  \[
  C(\mathcal{T}') = \sum_{i \in \mathcal{T}'} p_{i,\mathcal{B}} + \sum_{i \notin \mathcal{T}' \cup \mathcal{T}} p_{i,\mathcal{A}};
  \]
- The total energy of the schedule to \( \mathcal{T}' \) equals
  \[
  E(\mathcal{T}') = \sum_{i \in \mathcal{T}'} e_{i,\mathcal{B}} + \sum_{i \notin \mathcal{T}' \cup \mathcal{T}} e_{i,\mathcal{A}}.
  \]

The problem is to compute an allocation that minimize both the energy and the length of the schedule.

**Remark**

These two considered criteria are usually antagonistic since it takes more energy to perform a task faster.
Problem SCHED(p,e,C,E)

Input Execution times $p_{i,j}$ and energy values $e_{i,j}$ for $i \in \mathcal{T}$, $j \in \{A, B\}$; Values $C$ and $E$;

Output Is there exists $T' \subseteq T$ such that $C(T') \leq C$ and $E(T') \leq E$?

Theorem (Ait Aba et al. 2020)

$SCHED(p,e,C,E)$ is NP-complete problem.

The proof is based on a polynomial transformation from KNAPSACK.

Problem KNAPSACK(v,s,V,S)

Input $O = \{1, \ldots, n\}$, vectors $v_i$ and $s_i$ for $i \in O$; Values $S$ and $V$;

Output Is there exists $O' \subseteq O$ such that $\sum_{i \in O'} v_i \geq V$ and $\sum_{i \in O'} s_i \leq S$?
KNAPSACK(v,s,V,S) \propto SCHED(p,e,C,E)

- \mathcal{T} = \emptyset;
- \text{For any task } i \in \mathcal{T}, e_i,\mathcal{A} = 0 \text{ and } e_i,\mathcal{B} = s_i; \; E = S;
- \text{For any task } i \in \mathcal{T}, p_i,\mathcal{A} = v_1 + 1 \text{ and } p_i,\mathcal{B} = 1;

What is the right value of \( C \)?

The idea is to build the associated solution of SCHED(p,e,C,E) by setting \( \mathcal{T}' = \emptyset' \):

\[
C(\mathcal{T}') = \sum_{i \in \mathcal{T}} p_i,\mathcal{B} + \sum_{i \not\in \mathcal{T}} p_i,\mathcal{A} = \sum_{i \in \mathcal{T}} p_i,\mathcal{B} + (\sum_{i \in \mathcal{T}} p_i,\mathcal{A} - \sum_{i \not\in \mathcal{T}} p_i,\mathcal{A})
\]

Thus,

\[
C(\mathcal{T}') = \sum_{i \in \mathcal{T}} p_i,\mathcal{A} + \sum_{i \not\in \mathcal{T}} (p_i,\mathcal{B} - p_i,\mathcal{A}) = \sum_{i \in \mathcal{T}} p_i,\mathcal{A} - \sum_{i \not\in \mathcal{T}} v_i.
\]

So, we set \( C = \sum_{i \in \mathcal{T}} p_i,\mathcal{A} - V \) to obtain that \( C(\mathcal{T}) \leq \sum_{i \in \mathcal{T}} p_i,\mathcal{A} - V \).
The machine is composed by a chain of $m$ AI accelerators $1 \rightarrow 2 \rightarrow m$;

- The performance of each of them is characterized by a parameter $f_j$, $j \in \{1, \ldots, m\}$ to be determined;

- The execution time of $i$ on processor $j$ verifies $p_i(f_j) = \frac{p_i}{f_j}$;

- The energy for the execution of $i$ on processor $j$ is a non-decreasing function of $f_i$ denoted $e_i(f_j)$.

The idea is to schedule the tasks to the machine in a cyclic way, in order to execute sequentially several identical instances of the same chain.

- A feasible schedule executes the first tasks of the chain on the first processor, then the next one on the second, and so on ..;

- For any task $i \in T$, $\pi(i) \in \{1, \ldots, m\}$ is the processor that executes $i$.

- $\pi(1) = 1 \leq \pi(2) \leq \ldots \pi(n) \leq m$. 
A feasible solution of the scheduling problem is given by the couple \((\pi, f)\).

Two main criteria are considered:

- The period of the schedule is \(P(\pi, f) = \max_{j \in \{1, \ldots, m\}} \sum_{i, \pi(i) = j} p_i(f_j)\) is the maximum execution time of the tasks on a machine;
- The energy of the schedule is \(E(\pi, f) = \sum_{i \in \mathcal{T}} e_i(f_{\pi(i)})\).

**Theorem**

*For any upper bound \(P\) of the maximum period, finding a couple \((\pi, f)\) associated to a feasible solution that minimizes the energy without exceeding the periods bound can be done in time complexity \(O(n^4)\).*
Outline of the algorithm

Let us consider that $u \subseteq \mathcal{T}$ is the set of tasks performed by machine $j$.

- The set $u$ is feasible if $\sum_{i \in u} p_i(f_i) \leq P$, thus $f_j \geq P \cdot \sum_{i \in u} p_i$. We set $F(u) = P \cdot \sum_{i \in u} p_i$ as the minimum value for $f_j$;
- Let $U$ be the set of subsets of $\mathcal{T}$ composed by consecutive tasks. $|U| \leq n^2$.

Let us consider the labelled directed acyclic graph $\mathcal{G} = (V, A, \ell)$ built as follows:

- $V = U \cup \{s, p\}$; $s$ and $p$ are associated to $\emptyset$, thus $F(p) = F(s) = 0$;
- For any couple $(u, v) \in U^2$, the arc $(u, v) \in A$ if $u \cap v = \emptyset$ and there exists $i \in \{1, \ldots, n\}$ such that $i$ is the last task of $u$ and $i + 1$ the first task of $v$;
- For every $u \in U$ such that $1 \in u$, the arc $(s, u) \in A$; for every $u \in U$ such that $n \in u$, the arc $(u, p) \in A$;
- Each arc $a = (u, v) \in A$ is labelled by $\ell(a) = F(u)$. 
Outline of the algorithm

A feasible solution $S = (\pi, f)$ can be built from any path $\mu = s, u_1, \ldots, u_k, p$ from $s$ to $p$ as follows:

- For any task $i \in u_\alpha$, $\alpha \in \{1, \ldots, k\}$, $\pi(i) = \alpha$;
- For any $\alpha \in \{1, \ldots, k\}$, $f_\alpha = F(u_\alpha)$.

Moreover, any feasible solution is associated to a path from $s$ to $p$.

Thus, a feasible solution $S^* = (\pi^*, f^*)$ of minimum energy can be built from a shortest path $\mu^* = s, u_1^*, \ldots, u_k^*, p$ from $s$ to $p$.

Moreover, the minimum value of the energy is then $E(\pi^*, f^*) = \sum_{\alpha=1}^{k^*} F(u^*_\alpha)$.

We get an algorithm of time complexity $O(n^4)$ using Dijsktra’s shortest path algorithm.
Conclusion and perspectives

1. When a precise model for the energy is not available, we have to measure manually the power of the processing elements and to consider numerical values instead of equations;

2. The scheduling of a chain of task on a GPU/CPU platform considering as criteria the total length of the schedule and the total energy is an NP-complete problem;

3. The cyclic scheduling of a chain of task on a dedicated linear platform considering as criteria the period and the total energy is solvable in polynomial time.

Theoretical and practical perspectives are numerous:

1. Is there good approximation algorithm for the scheduling of a chain of task on a GPU/CPU platform? Is there Fixed-Parameter tractable algorithms? Which parameters...

2. Should we build a unique accelerator AI or split it into a chain of accelerators as suggested?