Overview

Table des matières

1 The context 1
2 Routing packets with fixed communication routes 1
3 Resolution of the “fluidified” problem 2
4 Building a schedule 4
5 Routing packets with freedom on the communication paths 7

1 The context

Platform

Platform : heterogeneous and distributed :
– processors with different capabilities ;
– communication links of different characteristics.

Applications

Application made of a very (very) large number of tasks, the tasks can be clustered into a finite number of types, all tasks of a same type having the same characteristics.

Principle

When we have a very large number of identical tasks to execute, we can imagine that, after some initiation phase, we will reach a (long) steady-state, before a termination phase.

If the steady-state is long enough, the initiation and termination phases will be negligible.

2 Routing packets with fixed communication routes

The problem

Problem : sending a set of message flows.

In a communication network, several flow of packets must be dispatched, each packet flow must be sent from a route to a destination, while following a given path linking the source to the destination.
Notations

– \((V, A)\) an oriented graph, representing the communication network.

– A set of \(n_c\) flows which must be dispatched.

– The \(k\)-th flow is denoted \((s_k, t_k, P_k, n_k)\), where
  – \(s_k\) is the source of packets;
  – \(t_k\) is the destination;
  – \(P_k\) is the path to be followed;
  – \(n_k\) is the number of packets in the flow.

  We denote by \(a_{k,i}\) the \(i\)-th edge in the path \(P_k\).

Hypotheses

– A packet goes through an edge \(A\) in a unit of time.

– At a given time, a single packet traverses a given edge.

Objective

We must decide which packet must go through a given edge at a given time, in order to minimize the overall execution time.

Lower bound on the duration of schedules

We call congestion of edge \(a \in A\), and we denote by \(C_a\), the total number of packets which go through edge \(a\) :

\[
C_a = \sum_{k \mid a \in P_k} n_k \quad C_{\text{max}} = \max_a C_a
\]

\(C_{\text{max}}\) is a lower bound on the execution time of any schedule.

\(C^* \geq C_{\text{max}}\)

A “fluid” (fractional) resolution of our problem will give us a solution which executes in a time \(C_{\text{max}}\).

3 Resolution of the “fluidified” problem

Fluidified (fractional) version : notations

Principle :

– we do not look for an integral solution but for a rational one.

– \(n_{k,i}(t)\) (fractional) number of packets waiting at the entrance of the \(i\)-th edge of the \(k\)-th path, at time \(t\).

– \(T_{k,i}(t)\) is the overall time used by the edge \(a_{k,i}\) for packets of the \(k\)-th flow, during the interval of time \([0; t]\).
Fluidified (fractional) version: writing the equations

1. Initiating the communications

\[ n_{k,1}(t) = n_k - T_{k,1}(t), \quad \text{for each } k \]

2. Conservation law

\[ n_{k,i+1}(t) = T_{k,i}(t) - T_{k,i+1}(t), \quad \text{for each } k \]

3. Resource constraints

\[ \sum_{(k,i) \mid a_{k,i} = a} T_{k,i}(t_2) - T_{k,i}(t_1) \leq t_2 - t_1, \forall a \in A, \forall t_2 \geq t_1 \geq 0 \]

4. Objective

\[ \text{MINIMIZE } C_{\text{frac}} = \int_0^\infty \mathbb{1} \left( \sum_{k,i} n_{k,i}(t) \right) dt \]

**Lower bound**

- \( n_{k,1}(t) = n_k - T_{k,1}(t), \) for each \( k \)
- \( n_{k,i+1}(t) = T_{k,i}(t) - T_{k,i+1}(t), \) for each \( k \)
- At any time \( t, \) \( \sum_{j=1}^i n_{k,j}(t) = n_k - T_{k,i}(t) \)
- For each edge \( a: \) \( \sum_{(k,i) \mid a_{k,i} = a} \sum_{j=1}^i n_{k,j}(t) = \sum_{(k,i) \mid a_{k,i} = a} n_k - \sum_{(k,i) \mid a_{k,i} = a} T_{k,i}(t) \geq C_a - t \)

As long as \( t < C_a, \) there are packets in the system.

Therefore, \( C_{\text{frac}} \geq \max_a C_a = C_{\text{max}} \)

**A candidate for the solution**

For \( t \leq C_{\text{max}} \)
- \( T_{k,i}(t) = \frac{n_k}{C_{\text{max}}} t, \) for each \( k \) and \( i. \)
- \( n_{k,1}(t) = n_k - T_{k,1}(t) = n_k - \frac{n_k}{C_{\text{max}}} t = n_k \left( 1 - \frac{t}{C_{\text{max}}} \right), \quad \forall k \)
- \( n_{k,i}(t) = 0, \) for each \( k \) and \( i \geq 2. \)

For \( t \geq C_{\text{max}} \)
- \( T_{k,i}(t) = n_k \)
- \( n_{k,i}(t) = 0 \)

This solution is a schedule of makespan \( C_{\text{max}}. \) We still have to show that it is feasible.
Checking the solution (for $t \leq C_{\text{max}}$)

1. $n_{k,1}(t) = n_k - T_{k,1}(t)$, for each $k$
   Satisfied by definition.

2. $n_{k,i+1}(t) = T_{k,i}(t) - T_{k,i+1}(t)$, for each $k$
   $T_{k,i}(t) - T_{k,i+1}(t) = \frac{n_k}{C_{\text{max}}}t - \frac{n_k}{C_{\text{max}}}t = 0 = n_{k,i+1}(t)$

3. $\sum_{(k,i) \mid a_{k,i} = a} T_{k,i}(t_2) - T_{k,i}(t_1) \leq t_2 - t_1, \forall a \in A, \forall t_2 \geq t_1 \geq 0$
   $\sum_{(k,i) \mid a_{k,i} = a} T_{k,i}(t_2) - T_{k,i}(t_1) = \sum_{(k,i) \mid a_{k,i} = a} \frac{n_k}{C_{\text{max}}}(t_2 - t_1) = \frac{C_a}{C_{\text{max}}}(t_2 - t_1) \leq t_2 - t_1$

4 Building a schedule

Definition of a round

– $\Omega$ ≈ duration of a round (will be defined later).

– $m_k$ : number of packets of $k$-th flow distributed in a single round.

$$m_k = \left\lfloor \frac{n_k \Omega}{C_{\text{max}}} \right\rfloor.$$

– $D_a = \sum_{(k,i) \mid a_{k,i} = a} 1 = |\{k | a \in P_k\}|$

$$D_{\text{max}} = \max_a D_a \leq n_c$$

– Period of the schedule : $\Omega + D_{\text{max}}$.

Schedule

During the time interval $[j(\Omega + D_{\text{max}}); (j + 1)(\Omega + D_{\text{max}})]$ :

The link $a$ forwards $m_k$ packets of the $k$-th flow if there exists $i$ such that $a_{k,i} = a$.

The link $a$ remains idle for a duration of :

$$\Omega + D_{\text{max}} - \sum_{(k,i) \mid a_{k,i} = a} m_k$$

(If less than $m_k$ packets are waiting in the entrance of $a$ at time $j(\Omega + D_{\text{max}})$, $a$ forwards what is available and remains idle longer.)
Feasibility of the schedule

\[
\sum_{(k,i)|a_{k,i}=a} m_k = \sum_{(k,i)|a_{k,i}=a} \left\lfloor \frac{n_k \Omega}{C_{\text{max}}} \right\rfloor 
\leq \sum_{(k,i)|a_{k,i}=a} \left( \frac{n_k \Omega}{C_{\text{max}}} + 1 \right) 
\leq \frac{C_a}{C_{\text{max}}} \Omega + D_a 
\leq \Omega + D_{\text{max}}
\]

Behavior of the sources

- \(N_{k,i}(t)\) : number of packets of the \(k\)-th flow waiting at the entrance of the \(i\)-th edge, at time \(t\).

- \(a_{k,1}\) sends \(m_k\) packets during \([0, \Omega + D_{\text{max}}]\).
  \(N_{k,1}(\Omega + D_{\text{max}}) = n_k - m_k\)

- \(a_{k,1}\) sends \(m_k\) packets during \([\Omega + D_{\text{max}}, 2(\Omega + D_{\text{max}})]\).
  \(N_{k,1}(2(\Omega + D_{\text{max}})) = n_k - 2m_k\)

- We let \(T = \left\lceil \frac{C_{\text{max}}}{\Omega} \right\rceil (\Omega + D_{\text{max}})\)
  \(N_{k,1}(T) \leq n_k - \frac{T}{\Omega + D_{\text{max}}} m_k \leq n_k - \frac{n_k \Omega}{C_{\text{max}}} \frac{C_{\text{max}}}{\Omega} = 0\)

Propagation delay

- \(a_{k,1}\) sends \(m_k\) packets during \([0, \Omega + D_{\text{max}}]\).
  \(N_{k,1}(\Omega + D_{\text{max}}) = n_k - m_k\)
  \(N_{k,i\geq 3}(\Omega + D_{\text{max}}) = 0\)

- \(a_{k,1}\) sends \(m_k\) packets during \([\Omega + D_{\text{max}}, 2(\Omega + D_{\text{max}})]\).
  \(N_{k,1}(2(\Omega + D_{\text{max}})) = n_k - 2m_k\)
  \(N_{k,3}(2(\Omega + D_{\text{max}})) = m_k\)
  \(N_{k,i\geq 4}(2(\Omega + D_{\text{max}})) = 0\)

- The delay between the time a packet traverses the first edge of the path \(P_k\) and the time it traverses its last edge is, at worst :
  \((|P_k| - 1)(\Omega + D_{\text{max}})\)

We let \(L = \max_k |P_k|\).
Makespan of the schedule

\[ C_{\text{total}} \leq T + (L - 1)(\Omega + D_{\text{max}}) \]
\[ = \left\lceil \frac{C_{\text{max}}}{\Omega} \right\rceil (\Omega + D_{\text{max}}) + (L - 1)(\Omega + D_{\text{max}}) \]
\[ \leq \left( \frac{C_{\text{max}}}{\Omega} + 1 \right) (\Omega + D_{\text{max}}) + (L - 1)(\Omega + D_{\text{max}}) \]
\[ = C_{\text{max}} + LD_{\text{max}} + \frac{D_{\text{max}}C_{\text{max}}}{\Omega} + L\Omega \]

The lower bound is minimized by \( \Omega = \sqrt{\frac{D_{\text{max}}C_{\text{max}}}{L}} \)

\[ C_{\text{total}} \leq C_{\text{max}} + 2\sqrt{C_{\text{max}}D_{\text{max}}L} + D_{\text{max}}L \]

Asymptotic optimality

\[ C_{\text{max}} \leq C^* \leq C_{\text{total}} \leq C_{\text{max}} + 2\sqrt{C_{\text{max}}D_{\text{max}}L} + D_{\text{max}}L \]

\[ 1 \leq \frac{C_{\text{total}}}{C_{\text{max}}} \leq 1 + 2\sqrt{\frac{D_{\text{max}}L}{C_{\text{max}}} + \frac{D_{\text{max}}L}{C_{\text{max}}}} \]

With \( \Omega = \sqrt{\frac{D_{\text{max}}C_{\text{max}}}{L}} \)

Resources needed

\[ \sum_{(k,i) | a_{k,i} = a, k \geq 2} m_k \leq \sum_{(k,i) | a_{k,i} = a, k \geq 2} \left( \frac{n_k}{C_{\text{max}}} \sqrt{\frac{D_{\text{max}}C_{\text{max}}}{L}} + 1 \right) \]
\[ \leq \sqrt{\frac{D_{\text{max}}C_{\text{max}}}{L}} + D_{\text{max}} \]

Conclusion

– We forget the initiation and termination phases
– Rational resolution of the steady-state
– Round whose size is the square-root of the solution:
  – Each round “loses” a constant amount of time
  – The sum of the waisted times increases less quickly than the schedule
– Buffers of size the square-root of the solution
5 Routing packets with freedom on the communication paths

Problem

- Same problem than previously, but the communication paths are not fixed.
- A set of \( n_c \) collection of packets which must be dispatched.
- Each collection of packets is dispatched through a set of flows (the packets of a same collection may follow different paths).
- \( n^{k,l} \) the total number of packets to be dispatched from \( k \) to \( l \).
- \( n^{k,l}_{i,j} \) : the total number of packets to be dispatched from \( k \) to \( l \) and which go through the edge \((i, j)\).

Congestion : 
\[
C_{i,j} = \sum_{(k,l) \mid n^{k,l} > 0} n^{k,l}_{i,j}
\]
\[
C_{\text{max}} = \max_{i,j} C_{i,j}.
\]

Writing the equations (1)

1. Initiating the communications
\[
\sum_{j \mid (k,j) \in A} n^{k,l}_{k,j} = n^{k,l}
\]

2. Receiving the messages sent
\[
\sum_{i \mid (i,l) \in A} n^{k,l}_{i,l} = n^{k,l}
\]

3. Conservation law
\[
\sum_{i \mid (i,j) \in A} n^{k,l}_{i,j} = \sum_{i \mid (j,i) \in A} n^{k,l}_{j,i} \quad \forall (k,l), j \neq k, j \neq l
\]

Writing the equations (2)

4. Congestion
\[
C_{i,j} = \sum_{(k,l) \mid n^{k,l} > 0} n^{k,l}_{i,j}
\]
5. Defining the objective

\[ C_{\text{max}} \geq C_{i,j}, \quad \forall i, j \]

6. Objective function

Minimiser \( C_{\text{max}} \)

Linear program in rational numbers : can be solved in polynomial time by any linear program solver.

Routing algorithm

1. Compute the optimal value \( C_{\text{max}} \) of the previous linear program.

2. Let \( \Omega \) be some value later defined.
   During the interval \([p\Omega, (p+1)\Omega]\), the edge \((i, j)\) forwards :

\[ m_{k,l}^{i,j} = \left\lfloor \frac{n_{k,l}^{i,j}\Omega}{C_{\text{max}}} \right\rfloor \]

packets which go from \( k \) to \( l \).

3. Starting at time :

\[ T \equiv \left\lceil \frac{C_{\text{max}}}{\Omega} \right\rceil \Omega \leq C_{\text{max}} + \Omega \]

we process the \( M \) remaining sequentially, which takes a time \( ML \) (at worst) where \( L \) is the maximal length of a simple path in the network.

The schedule is feasible

\[ \sum_{(k,l)} m_{k,l}^{i,j} \leq \sum_{(k,l)} n_{k,l}^{i,j} = \frac{n_{i,j}\Omega}{C_{\text{max}}} \leq \Omega \]

Makespan

- We define \( \Omega \) by : \( \Omega = \sqrt{C_{\text{max}}n_c} \).

- The total number of packets remaining in the network at time \( T \) is at worst :

\[ 2|A|\sqrt{C_{\text{max}}n_c} + |A|n_c \]

- The makespan is then

\[ C_{\text{max}} \leq C^* \leq C_{\text{max}} + \sqrt{C_{\text{max}}n_c} + 2|A|\sqrt{C_{\text{max}}n_c}|V| + |A|n_c|V| \]