Distributing the Heat Equation

Tom Cornebize Yassine Hamoudi

Sunday, December 7th 2014

1 Cellular automata

Question 1.

Lemma 1. *N^2 applications of function δ are necessary to compute X^t from X^{t-1}.*

Proof. Each cell $X^t_{i,j}$ needs one application of δ to be computed from $X^{t-1}_{i,j}$. There are N^2 cells, so N^2 applications of δ are needed. □

Property 2. *tN^2 applications of function δ are necessary to compute X^t on $[\![0, N-1]\!]^2$.*

Proof. X^t is obtained after t applications of $\delta^\dagger$ on X^0. Each application needs N^2 calls to δ according to lemma 1. The whole computation needs tN^2 applications of δ. □

Question 2. Let p^2 be the number of processors.

For the sake of simplicity, we will suppose that p divides N. Take $n = \frac{N}{p}$.

We divide the grid into square zones of size n. Each of this zones is given to one processor, which stores the data in its own memory and performs the computation of δ for all its cells. See figure 1 for an example.

At each step of computation, each processor updates its sub-matrix cells using a temporary sub-matrix that replaces the old one once the computation step is finished. Indeed, if we update the cells "in place", we overwrite values that are still necessary to compute other cells.

The computation of δ for the cells at the edges of the zones requires communication to retrieve the current states of their neighbours in other zones.

Question 3. We assume that X^t is given as an array $PREV$ of size $(n+2)\times(n+2)$, where $X^t_{i,j}$ is written in $PREV[i][j]$ and where $PREV[i][j]$ are dummy values for $i \in \{0, n+1\}$ or $j \in \{0, n+1\}$.

We also assume that the order of messages is preserved.

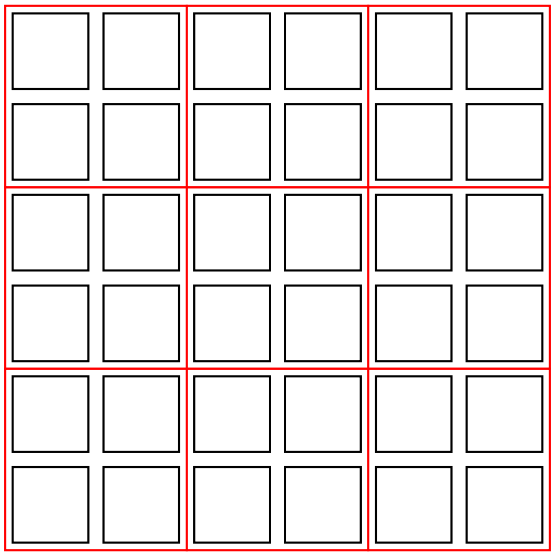
Let

$$\delta(PREV, i, j) = \delta\left(\begin{array}{|c|c|c|}\hline PREV[i-1,j-1] & PREV[i-1,j] & PREV[i-1,j+1] \\ \hline PREV[i,j-1] & PREV[i,j] & PREV[i+1,j+1] \\ \hline PREV[i+1,j-1] & PREV[i+1,j] & PREV[i+1,j+1] \\ \hline \end{array}\right)$$

We consider functions `Send_X` (resp. `Receive_X`) for `X = Top, Bottom, Left, Right` which sends (resp. receives) to (resp. from) the corresponding processor. We suppose that this function has a time cost of 1 and a communication cost of $L+b$ where L is the latency and b the bandwidth.

We also consider functions to send an entire row (resp. column) as one single message, to decrease the overall latency. For instance, `Send_Bottom_Row(n,PREV)` will send to the bottom processor the nth row, whereas `Send_Top_Row(0,PREV)` will receive from the top processor a row, which will be stored as the

Figure 1: Graphical representation of the topology for $N = 6$ and $p^2 = 9$.



0^{th} row. We suppose that this function has a time cost of n and a communication cost of $L + nb$ where L is the latency, b the bandwidth, and n the size of the row/column.

Algorithm 1: Stencil algorithm on a toric 2D grid

Input: PREV: array[0..n+1,0..n+1] of real
Output: NEXT: array[0..n+1,0..n+1] of real

```
/* Columns and row                                        */
Send_Left_Column(1,PREV)
Send_Right_Column(n,PREV)
Send_Top_Row(1,PREV)
Send_Bottom_Row(n,PREV)
Receive_Left_Column(0,PREV)
Receive_Right_Column(n+1,PREV)
Receive_Top_Row(0,PREV)
Receive_Bottom_Row(n+1,PREV)
/* Corners                                                */
Send_Top(PREV[1][0])
Send_Top(PREV[1][n+1])
Send_Bottom(PREV[n][0])
Send_Bottom(PREV[n][n+1])
Receive_Top(PREV[0][0])
Receive_Top(PREV[0][n+1])
Receive_Bottom(PREV[n+1][0])
Receive_Bottom(PREV[n+1][n+1])
/* Computation of δ                                       */
for i=1 to n do
    for j=1 to n do
        NEXT[i][j] = δ(PREV,i,j)
```

Time complexity: $8(n+1) + n^2\text{cost}(\delta) = 8(\frac{N}{p}+1) + \left(\frac{N}{p}\right)^2 \text{cost}(\delta) = \mathcal{O}\left(\left(\frac{N}{p}\right)^2\right)$ if $\text{cost}(\delta) = \mathcal{O}(1)$.

Communication complexity (one processor): $8(L + nb + L + 1) = 8(2L + \frac{N}{p}b + 1)$.

Communication complexity (all processors): $8p^2(2L + \frac{N}{p}b + 1) = 8(2p^2L + Npb + p^2)$.

2 Average automata

Question 4. See the implementation in `average.c`.

Question 5.

Property 3. *In the case of a p-average automaton, $\delta^\dagger$ is linear.*

Proof. Let $\delta^\dagger$ be the global transformation function of a *p-average automaton*. To prove that $\delta^\dagger$ is linear, it suffices to prove that the local transformation function δ is linear:

$$\delta\left(\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array}\right) = (1-p)\cdot e + p \cdot \frac{b+d+f+h}{4}$$

Let consider a real $k \in \mathbb{R}$, two local configurations $\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array}$ and $\begin{array}{|c|c|c|}\hline a' & b' & c' \\ \hline d' & e' & f' \\ \hline g' & h' & i' \\ \hline\end{array}$. We have:

$$\begin{aligned}
\delta\left(k\cdot\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array} + \begin{array}{|c|c|c|}\hline a' & b' & c' \\ \hline d' & e' & f' \\ \hline g' & h' & i' \\ \hline\end{array}\right) &= (1-p)\cdot(k\cdot e + e') + p\cdot\frac{(k\cdot b + b') + (k\cdot d + d') + (k\cdot f + f') + (k\cdot h + h')}{4} \\
&= k\cdot\left((1-p)\cdot e + p\cdot\frac{b+d+f+h}{4}\right) + (1-p)\cdot e' + p\cdot\frac{b'+d'+f'+h'}{4} \\
&= k\cdot\delta\left(\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array}\right) + \delta\left(\begin{array}{|c|c|c|}\hline a' & b' & c' \\ \hline d' & e' & f' \\ \hline g' & h' & i' \\ \hline\end{array}\right)
\end{aligned}$$

Thus δ is linear, and $\delta^\dagger$ too. □

Let's consider a configuration X. For $0 \leqslant i,j \leqslant N-1$ we define the matrix $E^{i,j}$ such that $E^{i,j}_{i,j} = 1$ and $E^{i,j}_{k,l} = 0$ otherwise. We obtain : $X = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{i,j}\cdot E^{i,j}$

Since $\delta^\dagger$ is linear, for all t :

$$\delta^{\dagger^t}(X) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{i,j}\cdot\delta^{\dagger^t}(E^{i,j})$$

Moreover :

$$\begin{aligned}
\delta^{\dagger^{2t}}(X) &= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{i,j}\cdot\delta^{\dagger^{2t}}(E^{i,j}) \\
&= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{i,j}\cdot\delta^{\dagger^t}(\delta^{\dagger^t}(E^{i,j})) \\
&= \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} X_{i,j}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\delta^{\dagger^t}(E^{i,j})_{k,l}\cdot\delta^{\dagger^t}(E^{i,j})
\end{aligned} \tag{1}$$

Property 4. *Equation 1 enables us to compute $\delta^{\dagger^t}(X)$ in time $O(logt)$.*

Proof. First of all, if $\delta^{\dagger^t}(E^{0,0})$ is already computed, we directly obtained by translation $\delta^{\dagger^t}(E^{i,j})$, for all i, j.

Let $T(t)$ the time needed to compute $\delta^{\dagger^t}(X)$. According to equation 1 and previous remark, we have:

$$\begin{aligned} T(2t) &= T(t) + O(1) \\ &= O(\log t) \end{aligned}$$

Thus, using the algorithm described by equation 1, we can compute $\delta^{\dagger^t}(X)$ in time $O(\log t)$. □

Property 5. *The time complexity $T'(t, N)$ in terms of both t and Nis $T'(t, N) = ...$*

Proof. □

3 Thermal reservoirs

Question 6.

Example.

Question 7.

Property 6. *For a p-average automaton with constants, $\delta^\dagger$ can be non-linear.*

Proof. Let's consider the following local configuration:

$$\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

We assume that 1 is a constant, but 0 not. We take $p = 0.5$.

We have:

$$\begin{aligned} \delta\left(\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}\right) &= \delta\left(\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}\right) \\ &= 0.5 \cdot 2 = 1 \end{aligned}$$

However:

$$\delta\left(\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}\right) + \delta\left(\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}\right) = 1 + 1 = 2$$

δ is not linear, and so $\delta^\dagger$ too. This result proves that $\delta^\dagger$ can be non-linear for a *p-average automaton* with constants. □