Design of Parallel and High Performance Computing
HS 2014
Torsten Hoefler, Markus Püschel
Department of Computer Science
ETH Zurich

Homework 11
Out: 2014-12-11
Revision: 1

Broadcast in the α-β-Model

The time taken to send a message of size s from one process to another is $T(s) = \alpha + s\beta$. If a process sends a message of size s at the time t it cannot send another message before $t + T(s)$.

In the lecture we have seen the analysis of a broadcast over a binary and a binomial tree. However, we can also define a k-ary as well as a k-nomial tree broadcast. In a k-ary tree brodcast every node forwards the received message to k children. A k-nomial tree is produced by forwarding the message to $k - 1$ children every round, until all processes are reached.

1. What is the runtime of a k-ary tree broadcast in the $\alpha\beta$ model if we assume small messages, i.e., $s = 1$?
2. What is the runtime of a k-nomial tree broadcast in the $\alpha\beta$ model if we assume small messages, i.e., $s = 1$?

Solution: Small Message k-ary Tree Broadcast in the $\alpha\beta$-Model

In a k-ary tree each non-leaf node has to send to k messages which takes this node $k \cdot (\alpha + s\beta)$ time units. Since we assume $s = 1$, and therefore do not do pipelining, the node can only start sending after he received the message from his parent. Since we are interested in the worst case we will now look at the node which is the last one to receive the message from his parent. Therefore the last node will finish at $k \cdot h \cdot (\alpha + s \cdot \beta)$ where h is the height of the tree.

The relationship between the height of a (full) k-ary tree and the number of processes (vertices in the tree) is as follows: On level i (starting at the root with $i = 0$) a k-ary tree has k^i vertices. So we can write the sum of a tree of the height h as

$$S(h) = \sum_{i=0}^{h} k^i = \frac{k^{h+1} - 1}{k - 1}$$

The closed form can be obtained as follows:

$$S(h) = \sum_{i=0}^{h} k^i = 1 + k + k^2 + k^3 \ldots k^h$$

if we multiply both sides of this equation by k we get

$$k \cdot S(h) = k + k^2 + k^3 + k^4 \ldots k^{h+1}$$

now we subtract the previous two equations from each other

$$(k - 1) \cdot S(h) = (k + k^2 + k^3 + k^4 \ldots k^{h+1}) - (1 + k + k^2 + k^3 \ldots k^h)$$

all but the first and last summand are cancelled out, and we are left with

$$(k - 1) \cdot S(h) = \ldots k^{h+1} - 1$$

which gives the closed form for $S(h)$ after division by $(k - 1)$ on both sides.

If we rewrite this equation we get $h = log_k(p(k - 1) + 1) - 1$, which makes the total runtime $k \cdot (log_k(p(k - 1) + 1) - 1) \cdot (\alpha + s \cdot \beta)$

Solution: Small Message k-nomial Tree Broadcast in the $\alpha\beta$-Model

In a k-nomial tree each non-leaf node has to send to $k-1$ messages per round which takes this node $(k-1)\cdot(\alpha+s\beta)$ time units. Since we assume $s = 1$, and therefore do not do pipelining, the node can only start sending after he received the message from his parent. Since we are interested in the worst case we will now look at the node which is the last one to receive the message from his parent. Therefore the last node will finish at $(k-1)\cdot h\cdot(\alpha+s\cdot\beta)$ where h is the height of the tree. The number of nodes in a k-nomial tree of height h is $p = k^h$. Therefore we get $(k-1)\cdot log_k(p)\cdot(\alpha+s\cdot\beta)$.

Communication Cost Models

1. What are the differences between the $\alpha\beta$ model, the LogP and the LogGP model?
2. Can you think of useful additions to those models? Why could they be inaccurate in practice?

Solution: Differences between models

In the $\alpha\beta$ model messages can not be received in parallel. The LogP and LogGP model split the cost for sending/receiving a messge into a part which is attributed to the CPU (o) and a part which is attributed to the network (L, g, and $s\cdot G$). The CPU and network cost can overlap. In the LogP model all messages are of the same size, bigger data items are sent as multiple messages. The LogGP model allows arbitrary size messages.

None of those models takes the topology of the network into account — in practice it is cheaper sending to a process running on a different core on the same CPU than it is to send over the network. Furthermore they ignore congestion of network links (which depends on the network topology). Also in practice modern network cards are parallel and pipelined — sending the same message twice (to different endpoints) is usually cheaper than sending different messages.

1 Strassen Matrix-Multiplication

In the lecture, we discussed recursive matrix multiplication

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

computed as

$$\begin{aligned} C_{11} &= A_{11}\cdot B_{11} + A_{12}\cdot B_{21} \\ C_{21} &= A_{21}\cdot B_{11} + A_{22}\cdot B_{21} \\ C_{12} &= A_{11}\cdot B_{12} + A_{12}\cdot B_{22} \\ C_{22} &= A_{21}\cdot B_{12} + A_{22}\cdot B_{22} \end{aligned}$$

which has the following communication cost (see lecture for derivation)

$$T_{\mathrm{MM}}(n,p) = O\left(\frac{n^2}{p^{2/3}}\cdot\beta\right) + O(\log(p)\cdot\alpha).$$

Strassen's algorithm reorganizes 2-by-2 block matrix multiplication and achieves a lower asymptotic computation cost complexity ($O(n^{log_2(7)})$ instead of $O(n^3)$):

$$
\begin{aligned}
M_1 &= (A_{11} + A_{22}) \cdot (B_{11} + B_{22}) \\
M_2 &= (A_{21} + A_{22}) \cdot B_{11} \\
M_3 &= A_{11} \cdot (B_{12} - B_{22}) \\
M_4 &= A_{22} \cdot (B_{21} - B_{11}) \\
M_5 &= (A_{11} + A_{12}) \cdot B_{22} \\
M_6 &= (A_{21} - A_{11}) \cdot (B_{11} + B_{12}) \\
M_7 &= (A_{12} - A_{22}) \cdot (B_{21} + B_{22}) \\
C_{11} &= M_1 + M_4 - M_5 + M_7 \\
C_{21} &= M_2 + M_4 \\
C_{12} &= M_3 + M_5 \\
C_{22} &= M_1 - M_2 + M_3 + M_6
\end{aligned}
$$

Derive the asymptotic communication cost of Strassen's algorithm on p processors using the α-β model. Does Strassen's algorithm have a higher flop/byte (computation per communication) ratio than regular standard matrix multiplication?

Solution: Strassen