# Two-Phase Broadcasting
# (PSC §2.4)

# Optimising a parallel algorithm

- Computation: well-balanced, little redundancy. Hence no room for improvement.
- Communication: every bit of communication is one bit too much. We can always try harder.

# Communication volume and balance

- The communication volume of an $h$-relation is the total number of data words communicated,

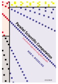
$$V = \sum_{s=0}^{p-1} h_{\mathrm{s}}(s) = \sum_{s=0}^{p-1} h_{\mathrm{r}}(s).$$

- $h_{\mathrm{s}}(s)$ is the number of data words sent by processor $P(s)$ and $h_{\mathrm{r}}(s)$ is the number received.
- Note that

$$V \leq \sum_{s=0}^{p-1} h = ph.$$

- An $h$-relation is balanced if

$$h = \frac{V}{p}.$$



# Communication imbalance

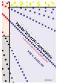
- The communication imbalance of an $h$-relation is

$$h - \frac{V}{p}.$$

- If an $h$-relation is balanced, so that

$$V = \sum_{s=0}^{p-1} h_{\mathrm{s}}(s) = ph,$$

  then $h_{\mathrm{s}}(s) = h$ for all $s$. (Because $h_{\mathrm{s}}(s) \leq h$.)
  Similarly, $h_{\mathrm{r}}(s) = h$ for all $s$.
- The reverse is also true: if $h_{\mathrm{s}}(s) = h$ for all $s$, then $V = ph$.
- Therefore, a balanced $h$-relation and a full $h$-relation are the same.



# $h_\mathrm{s} \neq h_\mathrm{r}$ implies communication imbalance

- If an $h$-relation is balanced, we have $h_\mathrm{s} = h_\mathrm{r}$, where $h_\mathrm{s} = \max_s h_\mathrm{s}(s)$ and $h_\mathrm{r} = \max_s h_\mathrm{r}(s)$.
- The reverse is not true: sending and receiving can have an equally overloaded processor, so that $h_\mathrm{s} = h_\mathrm{r}$, while the $h$-relation is still unbalanced, with $h \gg V/p$.
- $h_\mathrm{s} \neq h_\mathrm{r}$ implies that the communication is unbalanced.

# Communication imbalance in LU decomposition

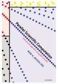
- Send cost in superstep (10), the row/column broadcast, assuming $M = N = \sqrt{p}$:

$$h_{\mathrm{s}} = R_{k+1}(N-1) + C_{k+1}(M-1) = 2R_{k+1}(\sqrt{p}-1).$$

- Receive cost in superstep (10):

$$h_{\mathrm{r}} = R_{k+1} + C_{k+1} = 2R_{k+1}.$$

- Large discrepancy: $h_{\mathrm{s}} \gg h_{\mathrm{r}}$. Balance for senders must be improved to reduce the communication cost.



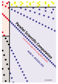
# Cause of the communication imbalance

$$\text{(10a)}\ \textbf{if}\ \phi_1(k) = t\ \textbf{then for all}\ i : k < i < n \wedge \phi_0(i) = s\ \textbf{do}$$
$$\text{put } a_{ik} \text{ in } P(s, *);$$

- The sending part of the broadcast of column $k$ is unbalanced: only the $\sqrt{p}$ processors in $P(*, \phi_1(k))$ send.
- The senders send $R_{k+1}(\sqrt{p} - 1) \approx n - k - 1$ elements.
- The receiving part is balanced: all processors receive $R_{k+1} \approx (n - k - 1)/\sqrt{p}$ elements, except the senders.
- Total contribution of (10) to LU cost is about

$$\sum_{k=0}^{n-1} 2(n-k-1)g = 2g \sum_{k=0}^{n-1} k = 2g(n-1)n/2 \approx n^2 g.$$

- This is a bottleneck vs. the computation cost $2n^3/3p$.



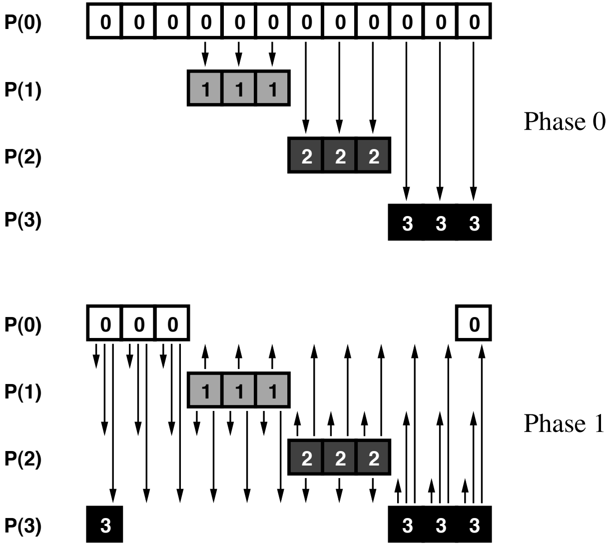
# One-phase broadcast of a vector

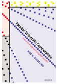
*input:* $\mathbf{x}$ : vector of length $n$, $\mathrm{repl}(\mathbf{x}) = P(0)$.
*output:* $\mathbf{x}$ : vector of length $n$, $\mathrm{repl}(\mathbf{x}) = P(*)$.
*call:* broadcast($\mathbf{x}, P(0), P(*)$).

{ Broadcast the vector. }
(0) **if** $s = 0$ **then for** $t := 0$ **to** $p - 1$ **do**
    **for** $i := 0$ **to** $n - 1$ **do**
        put $x_i$ in $P(t)$;

Note: $\mathrm{repl}(\mathbf{x}) = P(*)$ means that $\mathbf{x}$ is replicated such that each processor has a copy.

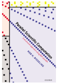
# Two-phase broadcast in blocks



# The two-phase idea

- First spread the data, then broadcast them. This lets every processor participate.
- This method is also used in the BitTorrent protocol, which splits a file to be distributed into small pieces and spreads these pieces among downloaders, who in turn make the pieces available for further distribution.
- Idea is similar to two-phase randomised routing (Valiant 1982): first send data to a randomly chosen intermediate location, then route them to their final destination. This avoids congestion.
- We don't need randomness here: in our regular problem, we can choose the intermediate location optimally and deterministically.

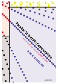


# Two-phase broadcast of a vector

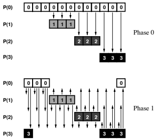
*input:* $\mathbf{x}$ : vector of length $n$, $\mathrm{repl}(\mathbf{x}) = P(0)$.
*output:* $\mathbf{x}$ : vector of length $n$, $\mathrm{repl}(\mathbf{x}) = P(*)$.
*call:* broadcast$(\mathbf{x}, P(0), P(*))$.

$b := \lceil n/p \rceil$;
{ Spread the vector. }
(0) **if** $s = 0$ **then for** $t := 0$ **to** $p - 1$ **do**
  **for** $i := tb$ **to** $\min\{(t+1)b, n\} - 1$ **do**
    put $x_i$ in $P(t)$;

{ Broadcast the subvectors. }
(1) **for** $i := sb$ **to** $\min\{(s+1)b, n\} - 1$ **do**
  put $x_i$ in $P(*)$;



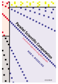
# Cost analysis of two-phase broadcast




- Phase 0 costs $(n-b)g$, where $b = \lceil n/p \rceil$ is the block size.
- Phase 1 costs $(p-1)bg$.
- Total cost of two-phase broadcast of a vector of length $n$ to $p$ processors is

$$T_{\text{broadcast}} = \left( n + (p-2) \left\lceil \frac{n}{p} \right\rceil \right) g + 2l \approx 2ng + 2l.$$

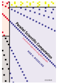
- Much less than the cost $(p-1)ng + l$ of a one-phase broadcast, except for large $l$.



# Two-phase broadcast in LU decomposition

$$
\begin{aligned}
&\text{broadcast}((a_{ik} : k < i < n \wedge i \bmod M = s),\\
&\qquad P(s, k \bmod N), P(s, *));\\
&\text{broadcast}((a_{kj} : k < j < n \wedge j \bmod N = t),\\
&\qquad P(k \bmod M, t), P(*, t));
\end{aligned}
$$

- Phase 0 of the row broadcast and Phase 0 of the column broadcast are done together in superstep (6).
- Phases 1 are done together in (7).
- Less modular, but more efficient.



# Optimisation: pivot value is already known

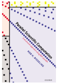
$$(8) \quad \textbf{if } \phi_0(k) = s \wedge \phi_1(k) = t \textbf{ then } \text{put } a_{kk} \text{ in } P(*, t);$$

- Delete old superstep (8), because

$$a_{kk} \text{ (after swap)} = a_{rk} \text{ (before swap)}.$$

  Pivot value $a_{rk}$ is already known locally.
- Divide immediately by $a_{rk}$ in new superstep (2) of Algorithm 2.8:

$$
\begin{aligned}
(2) \quad & \textbf{if } k \bmod N = t \textbf{ then} \\
& \qquad s_{\max} := \mathrm{argmax}(|a_{r_q,k}| : 0 \leq q < M); \\
& \qquad r := r_{s_{\max}}; \\
& \qquad \textbf{for all } i : k \leq i < n \wedge i \bmod M = s \wedge i \neq r \textbf{ do} \\
& \qquad\qquad a_{ik} := a_{ik}/a_{rk};
\end{aligned}
$$



# Optimisation: combine index and row swaps

(4) **if** $k \bmod M = s$ **then**

  **if** $t = 0$ **then** put $\pi_k$ as $\hat{\pi}_k$ in $P(r \bmod M, 0)$;

  **for all** $j : 0 \leq j < n \wedge j \bmod N = t$ **do**

    put $a_{kj}$ as $\hat{a}_{kj}$ in $P(r \bmod M, t)$;

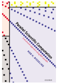
**if** $r \bmod M = s$ **then**

  **if** $t = 0$ **then** put $\pi_r$ as $\hat{\pi}_r$ in $P(k \bmod M, 0)$;

  **for all** $j : 0 \leq j < n \wedge j \bmod N = t$ **do**

    put $a_{rj}$ as $\hat{a}_{rj}$ in $P(k \bmod M, t)$;

Combining communication supersteps saves synchronisations.



# Optimisation: combine first and last superstep

**for** $k := 0$ **to** $n - 1$ **do**

(0) **if** $k \bmod N = t$ **then**

$r_s := \mathrm{argmax}(|a_{ik}| : k \leq i < n \wedge i \bmod M = s);$

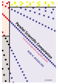
. . .

(0′) **for all** $i : k < i < n \wedge i \bmod M = s$ **do**

**for all** $j : k < j < n \wedge j \bmod N = t$ **do**

$a_{ij} := a_{ij} - a_{ik} a_{kj};$

- Combining the first and last superstep of the loop saves a synchronisation.
- In an implementation: no unnecessary `bsp_sync` at the end of the main loop.

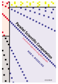


# Optimal aspect ratio $M/N$

- Two-phase broadcast reduces cost. Is $M = N$ still optimal?
- The cost of (6)/(7) is about $2(R_{k+1} + C_{k+1})g$. A bound is

$$\begin{aligned} R_{k+1} + C_{k+1} &< \left(\frac{n-k-1}{M} + 1\right) + \left(\frac{n-k-1}{N} + 1\right) \\ &= (n-k-1)\frac{M+N}{p} + 2, \end{aligned}$$

  which is indeed minimal for $M = N = \sqrt{p}$.
- The row and index swap in superstep (4) costs $(C_0 + 1)g$, where $C_0 = \lceil n/N \rceil$, so that larger values $N$ are preferred. Swap cost for $M = N$ is of same order as broadcast cost.
- Overall: $M = N$ close to optimal, but slight preference for $M < N$.



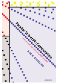
# Exact cost analysis

We need to compute sums of the form

$$\sum_{k=0}^{n-1} R_k = \sum_{k=0}^{n-1} \left\lceil \frac{n-k}{\sqrt{p}} \right\rceil = \sum_{k=1}^{n} \left\lceil \frac{k}{\sqrt{p}} \right\rceil .$$

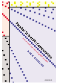
Lemma 2.9. Let $n, q \geq 1$ be integers with $n \bmod q = 0$. Then

$$\sum_{k=0}^{n} \left\lceil \frac{k}{q} \right\rceil = \frac{n(n+q)}{2q}, \qquad \sum_{k=0}^{n} \left\lceil \frac{k}{q} \right\rceil^2 = \frac{n(n+q)(2n+q)}{6q^2}.$$



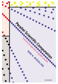
# Proof Lemma 2.9 (first part)

$$
\begin{aligned}
\sum_{k=0}^{n} \left\lceil \frac{k}{q} \right\rceil &= \left\lceil \frac{0}{q} \right\rceil + \left( \left\lceil \frac{1}{q} \right\rceil + \cdots + \left\lceil \frac{q}{q} \right\rceil \right) + \cdots \\
&\quad + \left( \left\lceil \frac{n-q+1}{q} \right\rceil + \cdots + \left\lceil \frac{n}{q} \right\rceil \right) \\
&= q \cdot 1 + q \cdot 2 + \cdots + q \cdot \frac{n}{q} \\
&= q \sum_{k=1}^{n/q} k \\
&= q \frac{n}{2q} \left( \frac{n}{q} + 1 \right) \\
&= \frac{n(n+q)}{2q}.
\end{aligned}
$$



# Total cost of LU decomposition

$$
\begin{aligned}
T_{\mathrm{LU}} &= \frac{2n^3}{3p} + \left(\frac{3}{2\sqrt{p}} - \frac{2}{p}\right) n^2 + \frac{5n}{6} \\
&\quad + \left( \left(\frac{3}{\sqrt{p}} - \frac{2}{p}\right) n^2 + \left(4\sqrt{p} - \frac{4}{\sqrt{p}} + \frac{4}{p} - 3\right) n \right) g \\
&\quad + 8nl \\
&\approx \frac{2n^3}{3p} + \frac{3n^2}{2\sqrt{p}} + \frac{3n^2 g}{\sqrt{p}} + 8nl.
\end{aligned}
$$



# Summary

- We have optimised our basic parallel LU decomposition algorithm by
  - performing two-phase broadcasting to spread the communication load evenly;
  - exploiting local information on the pivot value to avoid unnecessary communication;
  - reorganising the algorithm to combine supersteps, thus saving synchronisations.
- Cost analysis gives a diagnosis, such as $h_{\rm s} \gg h_{\rm r}$.
- The resulting LU decomposition is efficient if

$$\frac{2n^3}{3p} \geq \frac{3n^2 g}{\sqrt{p}} \quad \text{and} \quad \frac{2n^3}{3p} \geq 8nl.$$

- Equivalent to $n \geq \max\{4.5g, 2\sqrt{3l}\} \cdot \sqrt{p}$.