Bitonic Sorting

Mark Greenstreet, CpSc 448B, Term 1, 2011/12

Lecture Outline

Sorting

- The Bitonic Sorting Algorithm
- Revsort

Bitonic Sequences

- Definition
  - A sequence is bitonic iff it consists of an ascending sequence followed by a descending sequence or vice-versa.
  - More formally, $x_0, x_1, \ldots, x_{n-1}$ is bitonic iff

$$
\begin{array}{ll}
\multicolumn{2}{l}{\exists 0 \leq k < n-1.} \\
 & (\forall 0 \leq i < k.\ x_i \leq x_{i+1}) \wedge (\forall k \leq i < n-1.\ x_i \geq x_{i+1}) \\
\vee & (\forall 0 \leq i < k.\ x_i \geq x_{i+1}) \wedge (\forall k \leq i < n-1.\ x_i \leq x_{i+1})
\end{array}
$$

- Examples:
  - [0, 2, 4, 8, 10, 9, 7, 5, 3]
  - [10, 9, 7, 4, 0, 2, 4, 6, 9, 14]
  - [1, 2, 3, 4, 5]
  - []
  - but not [1, 2, 3, 1, 2, 3]

Properties of Bitonic Sequences

- Subsequences of bitonic sequences are bitonic:
  - If x is bitonic and has length n, and
  - if $0 \leq i_0 \leq i_1 \leq \ldots \leq i_{m-1} < n$,
  - then $[x_{i_0}, x_{i_1}, \ldots x_{i_{m-1}}]$ is bitonic.
  - This generalizes to $k-$tonic sequences, but we'll only need the bitonic version.
- If x is an up→down bitonic sequence, then so is reverse(x). Likewise for down→up sequences.

Bitonic Sort in Erlang

```
% sort(List, Up)
%     Sort List using the bitonic sorting algorithm.
%     If Up, sort the elements of List into ascending order.
%     Otherwise, sort them into descending order.
sort([], _) -> [];
sort([A], _) -> [A];
sort(X, Up) ->
   {X0, X1} = lists:split((length(X)+1) div 2, X),
   {Y0, Y1} = { sort(X0, Up), sort(X1, not Up) },
   merge(Y0 ++ Y1, Up).  % Note: Y0 ++ Y1 is bitonic
```

Example:

- Original list: `[24, 46, 2, 12, 98, 16, 67, 78]`.
- Split into two lists: `[24, 46, 2, 12]` and `[98, 16, 67, 78]`.
- Sort the first list ascending and the second descending:
  `[2, 12, 24, 46]` and `[98, 78, 67, 16]`
- Concatenate the two lists (bitonic result): `[2, 12, 24, 46, 98, 78, 67, 16]`
- Perform bitonic merge: `[2, 12, 16, 24, 46, 67, 78, 98]`

Bitonic Merge in Erlang

```
% merge(X, Up)
%     X is a bitonic sequence.
%      Return Y where Y is a list of the elements of X
%        in ascending order if Up is true and in descending order otherwise.
merge([A], _) -> [A]; % base case
merge(X, Up) -> % recursive case
    % split X into "even" and "odd" indexed sublists
    {X0, X1} = unshuffle(X),
    Y0 = merge(X0, Up), % recursively merge each sublist
    Y1 = merge(X1, Up),
    order([], shuffle(Y0, Y1), Up).   % compare-and-swap on even-odd pairs.
```

Example:

- List to merge: `[2, 12, 24, 46, 98, 78, 67, 16]`
- Unshuffle into even and odd lists: `[2, 24, 98, 67]` and `[12, 46, 78, 16]`.
- Recursively merge each list: `[2, 24, 67, 98]` and `[12, 16, 46, 78]`.
- Shuffle the merged sublists: `[2, 12, 24, 16, 67, 46, 98, 78]`.
- Compare-and-swap even-odd pairs: `[2, 12, 16, 24, 46, 67, 78, 98]`.

The order function

```
% order(Acc, List, Up) % compare-and-swap even-odd pairs of List
%     into ascending order if Up is true, and descending order otherwise.
%     The result is assembled in Acc.
%     Note, this is a tail-recursive implementation that reverses the order
%     of List in the process. That's OK because shuffle is tail recursive
%     as well and does another reverse that we cancel.
order(Acc, [], _) -> Acc;
order(Acc, [A], _) -> [A | Acc];
order(Acc, [A, B | T], Up) ->
   order(
      if
         (A == B) or ((A < B) == Up) -> [A, B | Acc];
         true -> [B, A | Acc]
      end,
   T, Up
).
```

Why Bitonic Merge Works

- Let X be a monotonically increasing sequence of 0's and 1's.
  - E.g. $X = [0, 0, 0, 0, 0, 0, 1, 1, 1, 1]$.
- Let Y be a monotonically decreasing sequence of 0's and 1's.
  - E.g. $Y = [1, 1, 1, 0, 0, 0, 0, 0, 0, 0]$.
- Let $Z = \text{concat}(X, Y)$. Note: Z is bitonic.
  - E.g.
$$\begin{aligned} Z &= [0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0], \\ &= [0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0], \\ Z_0 &= [0,0,0,1,1,1,1,0,0,0], \quad \text{\% } Z_0 \text{ is bitonic} \\ Z_1 &= [0,0,0,1,1,1,0,0,0,0], \quad \text{\% } Z_1 \text{ is bitonic.} \end{aligned}$$
- The number of 1's in Z_0 and Z_1 are nearly equal.
  - If the sequence of 1's in Z starts and ends at even-indexed elements, then $\text{NumberOfOnes}(Z_0) = \text{NumberOfOnes}(Z_1) + 1$.
  - If the sequence of 1's in Z starts and ends at odd-indexed elements, then $\text{NumberOfOnes}(Z_0) = \text{NumberOfOnes}(Z_1) - 1$.
  - Otherwise, $\text{NumberOfOnes}(Z_0) = \text{NumberOfOnes}(Z_1)$.
- At most one compare-and-swap is needed at the end.

For example...

- Continuing with our earlier example:

$$
\begin{aligned}
Z &= [0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0], \\
Z_0 &= [0,0,0,1,1,1,1,0,0,0], \\
Z_1 &= [0,0,0,1,1,1,1,1,1,1,0,0,0,0].
\end{aligned}
$$

- Recursively apply the merge procedure to Z_0 and Z_1 to get sorted lists, S_0 and S_1:

$$
\begin{aligned}
S_0 &= [0,0,0,0,0,0,1,1,1,1], \\
S_1 &= [0,0,0,0,0,0,0,1,1,1]
\end{aligned}
$$

- Shuffle S_0 and S_1 to get Y:

$$
Y = [0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1,1,1,1,1]
$$

- Continued on next slide.

continued example

- Coloring Y to highlight odd-even pairs:

$$\begin{aligned} Y &= [0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1,1,1,1,1], && \text{\% from prev. slide} \\ &= [0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,1,1,1,1,1], && \text{\% show even-odd pairs} \end{aligned}$$

- Note that there is one pair that needs to be swapped. Applying a compare-and-swap to each even-od pair yields:

$$S = [0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1]$$

- S is sorted.

More formally

- Let Z be a bitonic sequence of 0's and 1's
  - Let n be the length of Z. Index the elements of Z from 0 to $n-1$.
  - If Z is all 0's the bitonic network trivially sorts it.
  - Otherwise, let i be the index of the first 1 in Z and j be the index of the last 1.
- Let X be the even-indexed elements of Z:

$$\begin{aligned}\text{length}(X) &= \left\lceil \frac{n}{2} \right\rceil \\ x_k &= 0, \quad \text{if } 0 \leq k < \left\lceil \frac{i}{2} \right\rceil \text{ or } \left\lfloor \frac{j}{2} \right\rfloor < k < \left\lceil \frac{n}{2} \right\rceil \\ &= 1, \quad \text{if } \left\lceil \frac{i}{2} \right\rceil \leq k \leq \left\lfloor \frac{j}{2} \right\rfloor\end{aligned}$$

- Let $\tilde{X}$ be the sorted elements of X:

$$\begin{aligned}\tilde{x}_k &= 0, \quad \text{if } 0 \leq k < \left\lceil \frac{i}{2} \right\rceil + \left(\left\lceil \frac{n}{2} \right\rceil - \left\lfloor \frac{j}{2} \right\rfloor - 1 \right), \\ &= 1, \quad \text{otherwise}\end{aligned}$$

- Continued (next slide)

More formally (slide 2)

- Likewise, let Y be the odd-indexed elements of Z and $\tilde{Y}$ be the sorted elements of Y:

$$
\begin{aligned}
\text{length}(Y) &= \left\lfloor \tfrac{n}{2} \right\rfloor \\
y_k &= 0, \quad \text{if } 0 \le k < \left\lfloor \tfrac{i}{2} \right\rfloor \text{ or } \left\lceil \tfrac{j}{2} \right\rceil \le k < \left\lfloor \tfrac{n}{2} \right\rfloor \\
&= 1, \quad \text{if } \left\lfloor \tfrac{i}{2} \right\rfloor \le k \le \left\lceil \tfrac{j}{2} \right\rceil \\
\tilde{y}_k &= 0, \quad \text{if } 0 \le k < \left\lfloor \tfrac{i}{2} \right\rfloor + \left(\left\lfloor \tfrac{n}{2} \right\rfloor - \left\lceil \tfrac{j}{2} \right\rceil \right) \\
&= 1, \quad \text{otherwise}
\end{aligned}
$$

If n is even

- Let,

$$\begin{aligned} q_k &= \tilde{x}_{k/2}, && \text{if } k \text{ is even} \\ q_k &= \tilde{y}_{(k-1)/2}, && \text{if } k \text{ is odd} \\ r_k &= \min(q_k, q_{k+1}), && \text{if } k \text{ is even} \\ r_k &= \max(q_{k-1}, q_k), && \text{if } k \text{ is odd} \end{aligned}$$

  Claim: r_k is sorted. Need to show $\forall 1 \leq k < n . r_{k-1} \leq r_k$.

- If k is odd, the claim follows directly from the definition of r.
- If k is even, we need to show

$$\max(q_{k-2}, q_{k-1}) \leq \min(q_k, q_{k+1}) \quad \equiv \quad \max(\tilde{x}_{m-1}, \tilde{x}_{m-1}) \leq \min(\tilde{x}_m, \tilde{x}_m)$$

  where $m = k/2$.

- Because $\tilde{x}_{m-1} \leq \tilde{x}_m$ and $\tilde{x}_{m-1} \leq \tilde{x}_m$ it is sufficient to show $\tilde{x}_{m-1} \leq \tilde{x}_m$ and $\tilde{x}_{m-1} < \tilde{x}_m$.

n is even (continued)

- Equivalently, we can show $\tilde{x}_{m-1} = 1 \Rightarrow \tilde{x}_m = 1$ and $\tilde{x}_{m-1} = 1 \Rightarrow \tilde{x}_m = 1$.

$$
\begin{aligned}
& \tilde{x}_{m-1} = 1 \\
\Rightarrow \quad & m - 1 \geq \left\lceil \tfrac{i}{2} \right\rceil + \left(\left\lceil \tfrac{n}{2} \right\rceil - \left\lfloor \tfrac{j}{2} \right\rfloor - 1 \right) \\
\Rightarrow \quad & m \geq \left\lceil \tfrac{i}{2} \right\rceil + \left(\left\lceil \tfrac{n}{2} \right\rceil - \left\lfloor \tfrac{j}{2} \right\rfloor \right) \\
\Rightarrow \quad & m \geq \left\lfloor \tfrac{i}{2} \right\rfloor + \left(\left\lfloor \tfrac{n}{2} \right\rfloor - \left\lceil \tfrac{j}{2} \right\rceil \right) \\
\Rightarrow \quad & \tilde{x}_m = 1
\end{aligned}
$$

$$
\begin{aligned}
& \tilde{x}_{m-1} = 1 \\
\Rightarrow \quad & m - 1 \geq \left\lfloor \tfrac{i}{2} \right\rfloor + \left(\left\lfloor \tfrac{n}{2} \right\rfloor - \left\lceil \tfrac{j}{2} \right\rceil \right) \\
\Rightarrow \quad & m \geq \left\lfloor \tfrac{i}{2} \right\rfloor + \left(\left\lfloor \tfrac{n}{2} \right\rfloor - \left\lceil \tfrac{j}{2} \right\rceil \right) + 1 \\
\Rightarrow \quad & m \geq \left\lceil \tfrac{i}{2} \right\rceil + \left(\left\lfloor \tfrac{n}{2} \right\rfloor - \left\lfloor \tfrac{j}{2} \right\rfloor - 1 \right) \\
\Rightarrow \quad & m \geq \left\lceil \tfrac{i}{2} \right\rceil + \left(\left\lceil \tfrac{n}{2} \right\rceil - \left\lfloor \tfrac{j}{2} \right\rfloor - 1 \right), \qquad \left\lfloor \tfrac{n}{2} \right\rfloor = \left\lceil \tfrac{n}{2} \right\rceil \text{ because } n \text{ is even} \\
\Rightarrow \quad & \tilde{x}_m = 1
\end{aligned}
$$

If n is odd

- Let,

$$\begin{array}{rcll} q_k & = & \tilde{x}_{k/2}, & \text{if } k \text{ is even} \\ q_k & = & \tilde{y}_{(k-1)/2}, & \text{if } k \text{ is odd} \\ r_0 & = & q_0, & \\ r_k & = & \min(q_k, q_{k+1}), & \text{if } k \text{ is odd} \\ r_k & = & \max(q_{k-1}, q_k), & \text{if } k \text{ is even} \end{array}$$

  Claim: r_k is sorted. Need to show $\forall 1 \leq k < n . r_{k-1} \leq r_k$.

- Proof: similar to the n is even case. I'll write up the details for the posted slides.

- $\therefore$ bitonic merge is correct

Structure of a bitonic sorting network

Performance of bitonic sorting

Bitonic sort on real computers