OPTIMAL SELECTION ON NON-PARTITIONABLE SUPERPOSED PARALLEL BUSES

Bruce W. Arden
Toshio Nakatani

CS-TR-072-87

January 1987

Optimal Selection on Non-partitionable Superposed Parallel Buses

Bruce W. Arden

College of Engineering and Applied Science
University of Rochester
Rochester, N.Y. 14627

Toshio Nakatani

Department of Computer Science
Princeton University
Princeton, N.J. 08544

ABSTRACT

This paper presents an optimal selection algorithm on $n \times n$ *non-partitionable* superposed parallel buses. For arbitrary k $(1 \leq k \leq n^2)$, we can find the k-th smallest item in $O(n)$ time on $n \times n$ processors, where each processor has one data item, and is connected to row and column buses. This performance is shown to be optimal within a constant factor. This algorithm is *non-adaptive* in the sense that the algorithm does not depend on the data items to select. The algorithm can be adapted to a mesh-connected computer with the same asymptotic time performance.

January, 1987

Optimal Selection on Non-partitionable Superposed Parallel Buses

Bruce W. Arden

College of Engineering and Applied Science
University of Rochester
Rochester, N.Y. 14627

Toshio Nakatani

Department of Computer Science
Princeton University
Princeton, N.J. 08544

1. Introduction

Since the discovery of linear-time sequential selection algorithms (Blum, Floyd, Pratt, Rivest, and Tarjan[1973]), many investigators have considered the selection problem in different contexts. A lower bound for a sequential selection problem was given by Pratt and F. Yao[1973], and Hyafil[1976]. The upper bound was successively improved by Schonhage, Paterson, and Pippenger[1976], and Hyafil[1976]. Dobkin and Munro[1978] and Munro and Paterson[1980] studied time and space bounds for the selection problem. Johnson and Mizoguchi[1978] analyzed the selection problem on the multiset. Parallel and distributed algorithms for selection problems have also been considered on various models.

Chang and Roberts[1979], Hirschberg and Sinclair[1980], Franklin[1982], Dolev, Klawe, and Rodeh[1982], Peterson[1982], and Pachl, Korach, and Rotem[1982] have considered distributed maximum-finding algorithms on a ring of processors. Rodeh[1982], Santoro and Sidney[1982], and Chin and Ting[1985] studied a distributed median finding problem and its communication complexity. A more general framework for distributed

selection problem has also been examined.

Shrira, Francez, and Rodeh[1983] studied transformation from a sequential selection algorithm to a distributed algorithm in the context of arbitrary interconnection schemes. Frederickson[1983] analyzed the selection problem with regard to ring, mesh, and tree networks. Korach, Rotem, and Santoro[1984] were concerned with the distributed algorithms for finding median in the general network of computers and Zaks[1985] studied a distributed selection algorithm on a tree network. Akl[1984] and Gupta and Bhattacharjee[1984] concentrated on the parallel selection problem on the shared-memory SIMD model.

Levitan and Foster[1982] considered the maximum-finding problem using memories that permit only a single successful write but multiple reads. Stout[1983] studied the selection problem on a mesh with a shared global bus. Bokhari[1984] presented a maximum-finding algorithm also on a mesh with a shared global bus. Aggarwal[1986] extended his result to a mesh with k global buses. Kumar and Raghavendra[1985] and Stout[1986] also studied a selection problem on a mesh with multiple broadcast buses. Marberg and Gafni[1985] considered the selection problem on Multi-channel broadcast networks.

A. Yao[1980] and Wah and Chen[1984] viewed the problem in the context of networks. Cole and Yap[1985] and Ajtai, Komolos, Steiger, and Szemeredi[1986] were concerned with parallel selection algorithms and proved $O(\log\log n)$ bounds on Valiant's[1975] parallel comparison model.

In this paper, we derive lower and upper bounds for a selection problem on both a linear bus and a two dimensional grid of superposed parallel buses (SPB). For arbitrary k $(1 \leq k \leq n^2)$, we can find the k-th smallest item in $O(N)$ and $O(n)$ time on N processors connected by a shared bus and $n \times n$

processors connected by SPB, respectively. Both time bounds are asymptotically optimal within a constant factor.

The paper is organized as follows: In section 2, we summarize model of computation to make the basic assumptions clear. In section 3, we obtain lower and upper bounds for selection on a linear bus. In section 4, we derive a lower bound for selection on SPB. In section 5, we present an optimal algorithm for selection on SPB.

2. Model of Computation

The model of computation we assume is a SIMD (Single Instruction Stream Multiple Data Stream) machine. That is, we assume a parallel computer with N identical processors that execute the same instructions. Each processor is essentially a comparator with a pair of registers and a counter. The instructions can be either broadcast from the central control unit or distributed to each processor site beforehand. We assume that each processor initially has one data item for selection.

The interconnections between the processors are shared buses. We do not allow *bus-partitionability*. We assume unit delay for a wire of arbitrary length. That is, one bus transfer takes a unit of time. Only one processor is allowed to broadcast a data item to a shared bus and others may read it. The data items in each processor are not allowed to be modified. The only operations available are the comparison of two of the given data items, the update of the counter value, or the transmission of a single, unmodified data item from the collection. For the proof of a lower bound, infinite memory is allowed. For the proof of an upper bound, no extra memory is assumed for the linear case and no more than $O(\log^2 n)$ memory bits are assumed for the two dimensional case.

We consider two interconnection topologies: the linear bus and the two dimensional grid of superposed parallel buses. For the linear bus (Figure 2.1), we assume that N processors are connected by a shared bus. For the two dimensional grid interconnection (Figure 2.2), $N = n \times n$ processors, and each of the processors in every row and column are connected horizontally and vertically to a shared bus. Thus, there are two sets of n parallel buses. For deriving the lower bound on SPB, a processor is assumed to have double ports: one for the row bus and the other for the column bus. For upper bound on SPB, a processor is assumed to have only a single port and it can only access one of the row and column buses in one processor cycle.

2.1. Selection Problem

The k-th selection problem is well represented by the problem of finding a median. In the classical approach, the data items, or keys, are totally ordered with less-than-or-equal relation and then the k-th in the ordered sequence is selected. Here, k is called *the rank* of the selected item and clearly identical keys will have different but contiguous ranks. A possible alternative would be to arrange for all identical values to have the same rank, but if rank is to produce an ordered sequence, as opposed to being obtained from an ordered sequence, the distinct values of rank are preferable. The enumeration sort, referenced in this paper, is based on obtaining distinct values of rank.

3. Lower and Upper Bound for Selection on A Linear Bus

We prove that at least $\Omega(N)$ broadcast cycles are required to find the k-th smallest item from N processors connected by a shared bus.

Theorem 3.1: A specific selection from N items takes at least $N-1$ broadcast cycles on N processors connected by a shared bus.

Proof: First, we look at finding the maximum of N items. We use a kind of adversary argument. That is, whatever algorithm we may choose, there is always the worst case when finding maximum takes $N-1$ broadcast cycles. We use induction to prove the lower bound.

We note first that a processor becomes inactive if it compares a received, broadcast datum with its own datum and the broadcast value is larger. The resident datum is no longer a candidate for the maximum. An initial broadcast of the maximum would result in $N-1$ simultaneous comparisons and, in a sense, the completion of the selection in one cycle. Depending on broadcast order the number of concurrent comparisons can vary, but the worst case analysis below shows that $N-1$ broadcast cycles may be required.

Induction Hypothesis: For any $m<N$, finding maximum from m processors connected by a shared bus requires at least $m-1$ broadcast cycles.

Basis ($m=1$): Finding maximum from a single processor requires no broadcast and $N-1$, the minimum number of broadcasts required, is zero.

Induction Step: Any algorithm must do a comparison eventually after k broadcast cycles ($1 \leq k \leq N-1$). Whichever processors are chosen for k broadcast cycles, the worst case of broadcast items can be a collection of k smallest items (the smallest, the 2nd smallest, ..., the k-th smallest). Now, every processor does comparisons of any two data items it chooses among k broadcast items and its own item. No matter how many individual comparisons N processors may do in parallel using k broadcast items, no more than k items can be eliminated from the candidate list for maximum. That is, at least $N-k$ processors discard all the broadcast items as non-

candidates and only hold their own item as a candidate for maximum. Using induction hypothesis, it requires at least $N-k-1$ more broadcast cycles for the remaining $N-k$ processors to decide maximum. Therefore, finding maximum takes at least $N-1$ broadcast cycles.

For more general cases, such as finding the k-th smallest item, we can also find similar, worst case situations, as in the case of maximum. For example, we can pick up $k/2$ smallest and $k/2$ largest items for the worst case of selecting a median.□

For upper bound of selection on a linear bus, we present the following simple algorithm:

Theorem 3.2: N broadcast cycles are sufficient for selection of N items on N processors connected by a shared bus.

Proof: Each processor broadcasts its own data item starting with the left-most processor. Every time a new item is broadcast, every processor compares the broadcast item against its own item. If the broadcast item is smaller than its own item, processors increment their counter by one. If the broadcast item is equal to its own item and the broadcast item is from a processor whose id is smaller than its own id, then processors increment their counter by one. After all processors finish broadcasting their own data, each processor knows its rank from the counter value. Therefore, the k-th smallest item is known after N broadcast cycles.□

Corollary 3.1: A selection problem from N items takes $\Theta(N)$ broadcast cycles on N processors connected by a shared bus.

Proof: Immediate from Theorem 3.1 and 3.2.□

4. Lower Bound for Selection on SPB

We prove that at least $\Omega(n)$ broadcast cycles are required to find the kth smallest item from $n \times n$ grid of processors connected by SPB.

Theorem 4.1: A specific selection from $N = n^2$ $(n \geq 2)$ items takes at least $\lceil n/2 \rceil$ broadcast cycles on $N = n \times n$ processors connected by SPB.

Proof: We can use a similar argument to the one in Theorem 3.1 for a linear bus. First, we look at finding the maximum of N items. We use induction to prove the lower bound.

Induction Hypothesis: For any $m < N$, finding the maximum from m processors connected by a $n \times n$ grid of SPB requires at least $\lceil m/2n \rceil$ broadcast cycles.

Basis $(m = 2)$: Finding the maximum from two processors connected by a $n \times n$ grid of SPB requires one broadcast cycle and $\lceil m/2n \rceil$, the minimum number of broadcasts required, is one.

Induction Step: Any algorithm must do a comparison eventually after k broadcast cycles $(1 \leq k \leq n-1)$. Whichever processors are chosen for k broadcast cycles, the worst case when searching for a maximum is to broadcast the $2kn$ smallest items. Every processor receives at most $2k$ broadcast items. This is because, for every broadcast cycle, at most $2n$ items can be broadcast on the total of $2n$ parallel buses and every processor receives at most two broadcast items. Now, every processor does comparisons of any two data items it chooses among $2k$ broadcast items and its own item. No matter how many individual comparisons $N = n^2$ processors may do in parallel, using $2k$ broadcast items, no more than $2kn$ items can be eliminated from the candidate list for maximum. That is, at least $n^2 - 2kn$ processors discard all the broadcast items as non-candidates and remain active to hold their own item as a candidate for maximum. Using induction hypothesis, it

requires at least $n/2-k$ more broadcast cycles for the remaining n^2-2kn processors to determine the maximum. Therefore, finding maximum takes at least $\lceil n/2 \rceil$ cycles.

For more general cases, such as finding the k-th smallest item, we can also find the worst case situations as in the case of maximum. For example, we can pick up kn smallest and kn largest items for the worst case of median finding using a similar argument.□

5. Upper Bound for Selection on SPB

We use row sort and column sort as basic operations for selection. For this purpose, enumeration sort (Arden and Nakatani[1987]) is used for optimal sorting on a linear bus. After an iteration of row and column sort, a grid of $n \times n$ items forms *a lattice* (Abbott[1969]). That is, it forms a partially ordered set and there always exist least upper bound (join or union) and greatest lower bound (meet or intersection) for any two items. The basic strategy to find the k-th smallest item is based on the fact that large a fraction of items can be eliminated as non-candidates for the k-th smallest item after row and column sorts.

First of all, we can find the maximum and minimum at the upper-left corner and the lower-right corner, respectively. In other words, we can completely eliminate non-candidates for maximum and minimum by one iteration of row and column sorts. In contrast, to find the median, the largest fraction must remain as candidates. However, even for this worst case, we can eliminate at least one quarter of all the items after one iteration of row and column sorts. In Figures 5.1a and 5.1b, we show a 8×8 grid of processors after row and column sort. The squares in grey color represent non-candidates for k-th smallest item ($1 \leq k \leq 12$) and therefore they can be

discarded.

More precisely, after row and column sorts, we can eliminate the subset, $S(n^2,k)=\{(x,y)$: $xy>k$ or $(n-x)(n-y)>n^2-k+1\}$ as non-candidates for the k-th smallest item from n^2 items, $\{(x,y)$: $1\leq x,y\leq n\}$. In general, the cardinality of $S(n^2,k)$ has following property:

$$|S(n^2,k)|\geq|S(n^2,\frac{n^2}{2})|>n^2\cdot(1-\log_e 2)>0.3n^2>\frac{n^2}{4}$$

For example $(n\geq 5)$,

$$|S(n^2,1)|=|S(n^2,n^2)|=n^2-1$$

$$|S(n^2,2)|=|S(n^2,n^2-1)|=n^2-2$$

$$|S(n^2,3)|=|S(n^2,n^2-2)|=n^2-4$$

$$|S(n^2,4)|=|S(n^2,n^2-3)|=n^2-7$$

$$|S(n^2,5)|=|S(n^2,n^2-4)|=n^2-9$$

First, we describe the algorithm informally using the example (Figure 5.2): We consider the problem of finding median on the 8×8 grid of processors connected by SPB. After all rows are sorted and then all columns are sorted, 26 items can be discarded as non-candidates for median and total of $64-26=38$ items remain active for a candidate. The active items are permuted to form a smaller grid of size 7×6. After a row and column sort on the new grid, 13 items can be discarded as non-candidates and total of $38-13=25$ items remain active. The active items are permuted to form a smaller grid of size 5×5. After row and column sort on the new grid, 12 items can be discarded and total of $25-12=13$ items remain active. The active items are permuted to form a grid of size 4×4. After row and column sort, 6 items can be discarded and total of 7 items remain active. After the number of active items is fewer than n (8 for this example), the active items are permuted to a single row and row sort is performed. Then, we can pick

the median at the center of the row.

It is important to note that the positions of discarded items, permutations, and the number of iterations do not depend on the specific data items. In other words, the algorithm is *non-adaptive*. Therefore, this information can be precalculated and stored in the memory of each processor. That is, at the beginning of each iteration, the i-th processor reads its new identification, id_i, the mask, $mask_i$, and the permutation, p_i from the memory. After row and column sort, it inactivates itself depending on its $mask_i$, sends its own item to the destination processor based on the permutation, p_i, and updates the iteration count by one and checks it against the total iteration number, *upto*.

The $O(n)$ *non-adaptive* selection algorithm, *Selection*(n, k), for $N=n^2$ data items on $N=n\times n$ processors connected by SPB appears below.

procedure *Selection*(n, k);

begin

read the total iteration value, *upto*, from the memory.

for $i=1$ to *upto* **do begin**

read the size of grid, n_i, identification, id_i, the mask, $mask_i$, and permutation, p_i.

for each row r ($r=0,1,\cdots,n_i-1$) in parallel **do**

Enumeration-Sort(n_i);

for each column c ($c=0,1,\cdots,n_i-1$) in parallel **do**

Enumeration-Sort(n_i);

if $mask_i=1$ **then**

active←*off*;

Permutation(p_i);

end

end;

Theorem 5.1: The algorithm *Selection*(n, k) takes $O(n)$ time.

Proof: One iteration of *Selection*(n_i, k) takes $O(n_i)$. This is because row or column sort takes $2n_i$ (Arden and Nakatani[1987]) and permutation takes at most $2n_i$ (Arden and Nakatani[1986c]). The number of iterations is at most $O(\log n)$ because $n_{i+1} < \frac{\sqrt{3}}{2} \cdot n_i$. That is, we have the following recurrence equation for the time complexity, $T_S(n)$:

$$T_S(n_i) < T_S(n_{i+1}) + 6n_i$$

By solving this recurrence equation, we obtain $T_S(n) < 12 \cdot (2+\sqrt{3}) \cdot n$.□

Corollary 5.1: A selection problem from $N = n^2$ items takes $\Theta(n)$ time on $N = n^2$ processors connected by SPB.

Proof: Immediate from Theorem 4.1 and Theorem 5.1.□

6. Concluding Remarks

We obtained lower and upper bounds for the selection problem on both a linear bus and a two dimensional grid of superposed parallel buses. The optimal selection algorithm for SPB in this paper can be adapted to a mesh-connected computer with the same asymptotic time performance. Enumeration sorting for row and column sort can be replaced by even-odd transposition sort on a mesh with the same time performance (in the restricted SIMD model) (Thompson and Kung[1977]). Permutation can be performed on a mesh with the same asymptotic time performance but with slightly larger constant factor.

As a closely related subject, we studied the sorting problem on SPB. There is a mismatch between the $O(n)$ lower bound and the $O(n\log\log n)$ upper bound for non-partitionable SPB (Arden and Nakatani[1987]). But, if we allow bus-partitionability, bitonic sorting on SPB with optimal $O(n)$ time performance can be implemented (Arden and Nakatani[1986d]). This is based on k-way bitonic sort (Arden and Nakatani[1986b]). However, bus-partitionability does not always improve the performance of algorithms on SPB. For example, the time for some permutations can not be reduced with partitionable buses (Arden and Nakatani[1986a and 1986c]).

An interesting question is how much the performance of a selection on SPB with partitionable buses can be improved. For example, if we allow bus-partitionability, a maximum-finding problem can be solved in $O(\log N)$ and $O(\log n)$ on N processors connected by a single shared bus and a $n \times n$ grid of processors connected by SPB, respectively. For the more general selection problem, finding the k-th smallest item can be performed in $O(\log^3 N)$ and $O(\log^3 n)$ on N processors connected by a single shared bus and a $n \times n$ grid of processors connected by SPB, respectively. This is based on the similar approach to Munro and Paterson[1980] and Frederickson[1983]. However, lower bounds for the general selection problem with partitionable buses are still unknown.

It is also interesting to alter the model of computation. For example, if we allow multiple writes but only one successful write to a bus, then a selection problem can be solved in average $O(\log N)$ time on N processors connected by a non-partitionable, shared bus (Levitan and Foster[1982]). Furthermore, if the range of keys is known, then selection can be done in $O(\log N)$ time on N processors connected by a non-partitionable, shared bus (Levitan and Foster[1982]).

References

Abbott, J. C. [1969]. *Sets, Lattices, and Boolean Algebras,* Allyn and Bacon, Inc., Boston.

Aggarwal, A. [1986]. "Optimal bounds for finding maximum on arrays of processors with k global buses," *IEEE Trans. on Computers,* C-35: 1, pp. 62-64.

Ajtai, M., J. Komlos, W. L. Steiger and E. Szemeredi [1986]. "Deterministic selection in O(loglogN) parallel time," *Proc. of 18th ACM Symposium on Theory of Computing,* pp. 188-195.

Akl, S. G. [1984]. "An optimal algorithm for parallel selection," *Information Processing Letter,* 19, pp. 47-50.

Arden, B. and T. Nakatani [1986a]. "Permutations on superposed parallel buses," Technical Report CS-TR-024-86, Department of Computer Science, Princeton University.

Arden, B. and T. Nakatani [1986b]. "K-way bitonic sort," Technical Report CS-TR-040-86, Department of Computer Science, Princeton University.

Arden, B. and T. Nakatani [1986c]. "Optimal permutations on superposed parallel buses," Technical Report CS-TR-060-86, Department of Computer Science, Princeton University.

Arden, B. and T. Nakatani [1986d]. "Bitonic sorting on superposed parallel buses," Technical Report CS-TR-063-86, Department of Computer Science, Princeton University.

Arden, B. and T. Nakatani [1987]. "Improved upper bound of sorting on non-partitionable superposed parallel buses," Technical Report CS-TR-071-87, Department of Computer Science, Princeton University.

Blum, M., R. Floyd, V. Pratt, R. Rivest, and R. Tarjan [1973]. "Time bounds for selection," *J. of Computer and System Sciences,* 7, pp. 448-461.

Bokhari, S. H. [1984]. "Finding maximum on an array processor with a global bus," *IEEE Trans. on Computers,* C-33:2, pp. 133-139.

Chang, E. and R. Roberts [1979]. "An improved algorithm for decentralized extrema-finding in circular configurations of processors," *Comm. of ACM,* 22, pp. 281-283.

Chin, F. and H. F. Ting [1985]. "A near-optimal algorithm for finding the median distributively," *Proc. of 5th IEEE International Conf. on Distributed Computing Systems,"* pp. 459-465.

Cole, R. and C. K. Yap [1985]. "A parallel median algorithm," *Information Processing letters,* 20, pp. 137-139.

Dobkin, D. P. and J. I. Munro [1978]. "Time and space bounds for selection problems," *Proc. of 5th International Colloquium on Automata, languages and Programming,* Udine, Italy.

Dolev, D., M. Klawe, and M. Rodeh [1982]. "An O(NlogN) unidirectional distributed algorithm for extrema finding in circles," *J. of Algorithms,* 3, pp. 245-260.

Franklin, R. [1982]. "On an improved algorithm for decentralized extrema finding in circular configurations of processors," *Comm. of ACM,* 25, pp. 336-337.

Frederickson, G. N. [1983]. "Tradeoffs for selection on distributed networks," *Proc. of 2nd Annual ACM Symposium of Principles of Distributed Computing,* pp. 154-160.

Gale, D. and R. M. Karp [1972]. "A phenomenon in the theory of sorting," *J. of Computer and System Sciences,* 6, pp. 103-115.

Gupta, P. and G. P. Bhattacharjee [1984]. "A parallel selection algorithm," *BIT,* 24, pp. 274-284.

Hirschberg, D. and J. Sinclair [1980]. "Decentralized extrema-finding in circular configurations of processors," *Comm. of ACM,* 23, pp. 627-628.

Hyafil, L. [1976]. "Bounds for selection," *SIAM J. of Computing* 5:1, pp. 109-114.

Johnson, D. B. and T. Mizoguchi [1978]. "Selecting the k-th element in X+Y and X1+X2+...+Xm," *SIAM J. of Computing,* 7:2, pp. 147-153.

Korach, E., D. Rotem, and N. Santoro [1984]. "Distributed algorithms for finding centers and medians in networks," *ACM Trans. on Programming Languages and Systems,* 6:3, pp. 380-401.

Knuth, D. E. [1973]. *The Art of Computer Programming, Vol 3: Sorting and Searching.* Reading, M.A., Addison-Wesley.

Kumar, V. K. P. and C. S. Raghavendra [1985]. "Array processor with multiple broadcasting," *Proc. of 12th Annual Symposium on Computer Architecture,* pp. 2-10.

Levitan, S. P. and C. C. Foster [1982]. "Finding an extremum in a network," *Proc. of 9th Annual Symposium on Computer Architecture,* pp. 321-325.

Marberg, J. M. and E. Gafni [1985]. "sorting and selection in multi-channel broadcast networks," *Proc. of International Conference on Parallel Processing,* pp. 846-850.

Munro, J. I. and M. S. Paterson [1980]. "Selection and sorting with limited storage," *Theoretical Computer Science,* 12, pp. 315-323.

Pachl, J., E. Korach, and D. Rotem [1982]. "A technique for proving lower bounds for distributed maximum-finding algorithms," *Proc. of 14th Annual ACM Symposium on Theory of Computing,* pp. 378-382.

Peterson, G. L. [1982]. "An O(NlogN) unidirectional algorithm for the circular extrema problem," *ACM Trans. on Programming Languages and Systems,* 4:4, pp. 758-762.

Pratt, V. and F. Yao [1973]. "On lower bounds for computing the i-th largest element," *Proc. of 14th Annual IEEE Symposium on Switching and Automata theory,* , pp. 70-81.

Preparata, F. P. [1978]. "New parallel-sorting schemes," *IEEE Trans. on Computers,* C-27:7, pp. 669-673.

Rodeh, M. [1982]. "Finding the median distributively," *J. of Computer and System Sciences,* 24, pp. 162-166.

Santros, N. and J. B. Sidney [1982]. "Order statistics on distributed sets," *Proc. of 20th Allerton conf. on Communication, Control, and Computing,* pp. 251-256.

Schonhage, A., M. Paterson, and N. Pippenger [1976]. "Finding the Median," *J. of Computer and System Sciences,* 13, pp. 184-199.

Shrira, L., N. Francez, and M. Rodeh [1983]. "Distributed k-selection: from a sequential to a distributed algorithm," *Proc. of 2nd Annual ACM Symposium of Principles of Distributed Computing,* pp. 143-153.

Stout, Q. F. [1983]. "Mesh-connected computers with broadcasting," *IEEE Trans. on Computers,* C-32: 9, pp. 826-830.

Stout, Q. F. [1986]. "Meshes with multiple buses," *Proc. of 27th Annual IEEE Symposium on Foundations of Computer Science,* pp. 264-273.

Thompson, C.D. and H. T. Kung [1977]. "Sorting on a mesh-connected parallel computers," *Comm. ACM* 20:4, pp. 263-271.

Valiant, L. [1975]. "Parallelism in comparison problems," *SIAM J. of Computing,* 4:3, pp. 348-355.

Wah, B. W. and K. Chen [1984]. "A partitioning approach to the design of selection networks," *IEEE Trans. on Computers,* C-33:3, pp. 261-268.

Yao, A. C. C. [1980]. "Bounds on selection networks," *SIAM J. of Computing,* 9, pp. 566-582.

Zaks, S. [1985]. "Optimal distributed algorithms for sorting and ranking," *IEEE Trans. on Computers,* C-34:4, pp. 376-379.

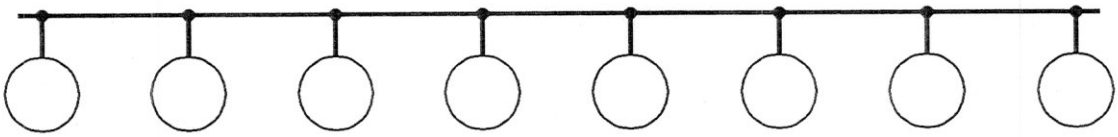


Figure 2.1: A linear bus ($N = 8$)

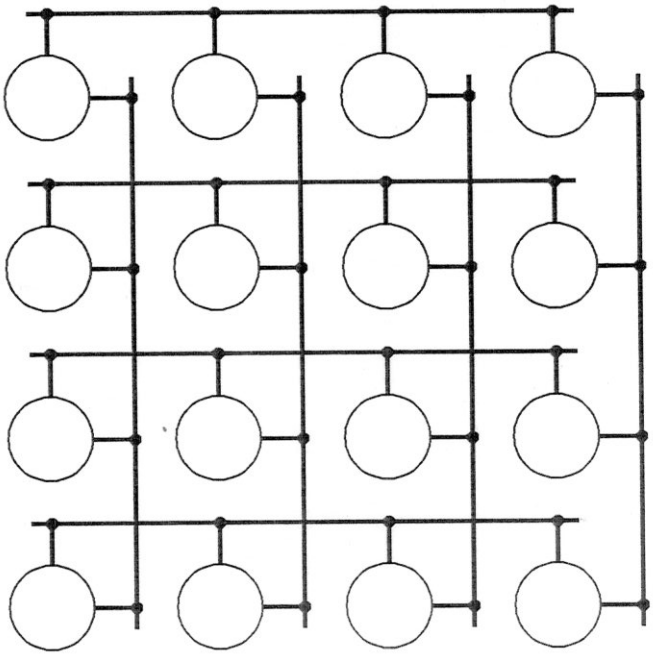
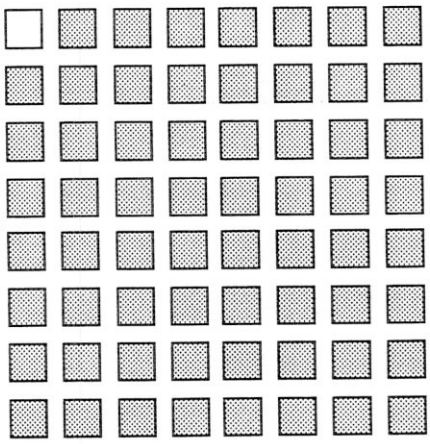
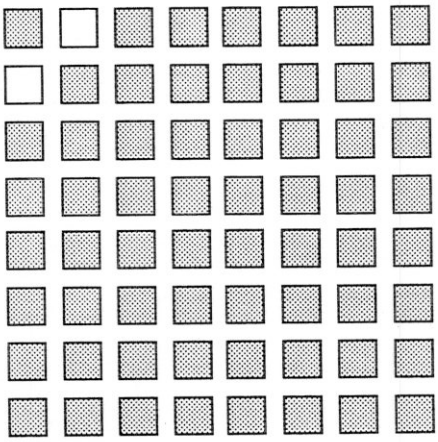


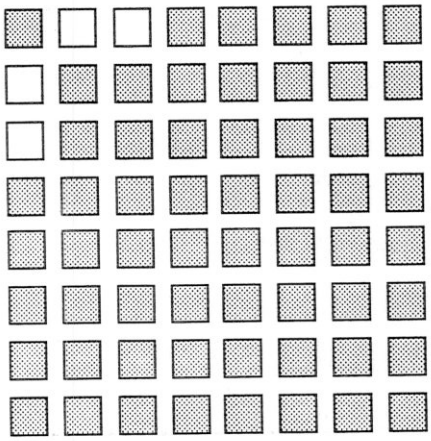
Figure 2.2: A square grid of superposed parallel buses ($N = 4x4$)



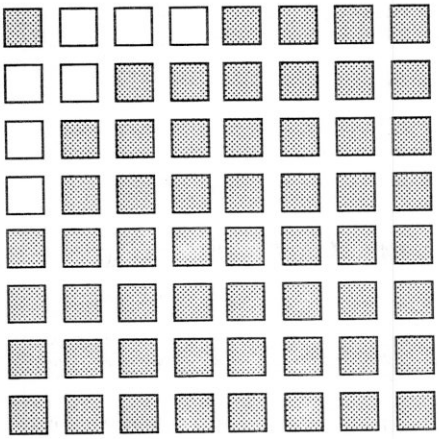
The smallest



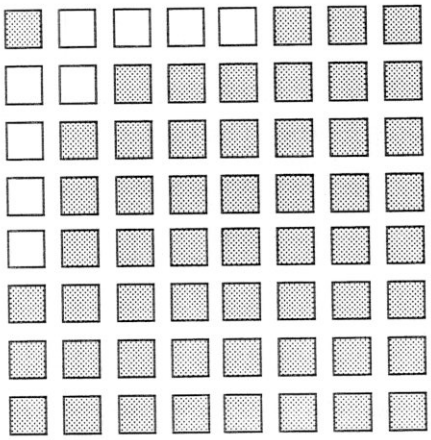
2-nd smallest



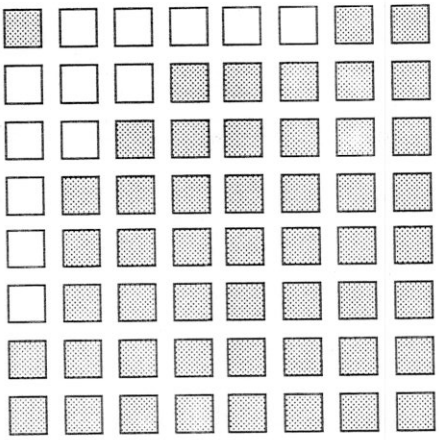
3-rd smallest



4-th smallest

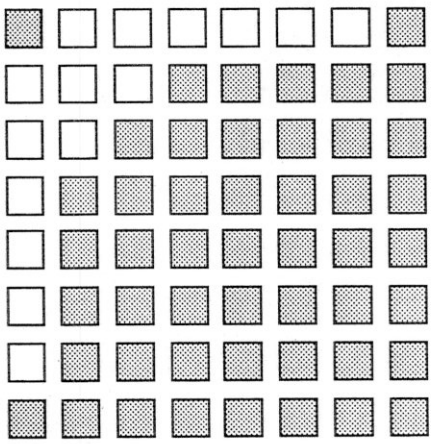


5-th smallest

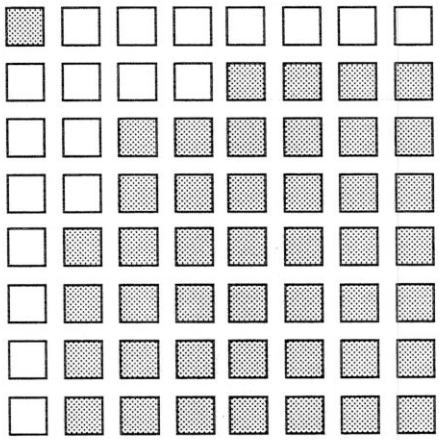


6-th smallest

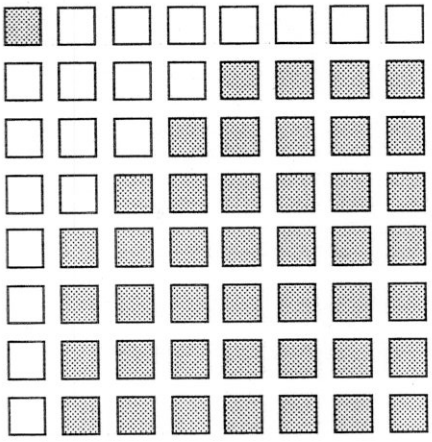
Figure 5.1a: The candidates for the *k*-th Smallest item ($1 \le k \le 6$)



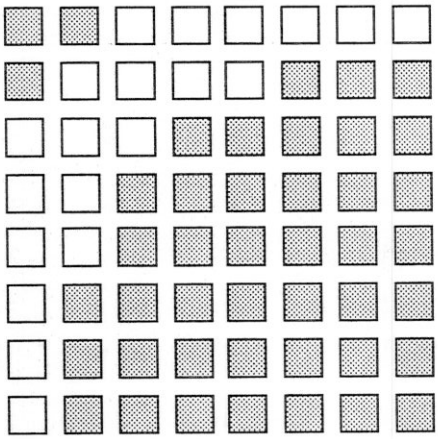
7-th smallest



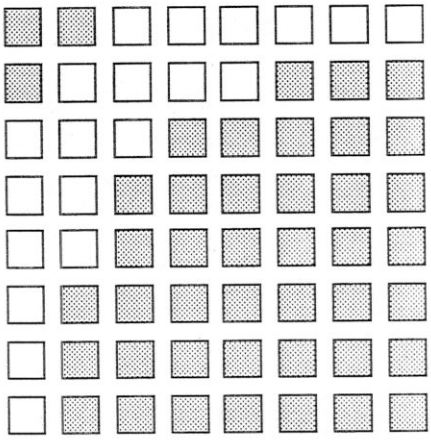
8-th smallest



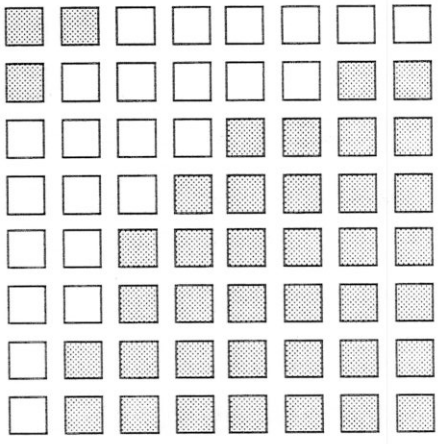
9-th smallest



10-th smallest

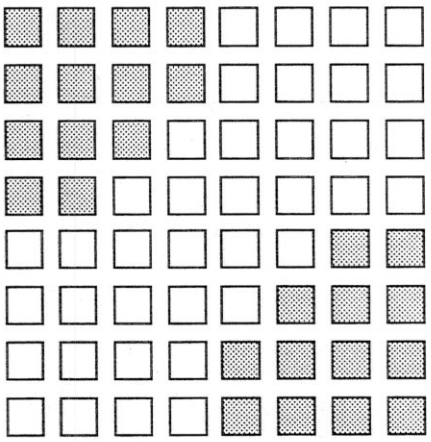


11-th smallest

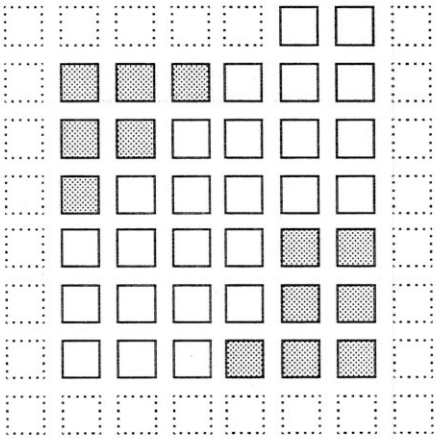


12-th smallest

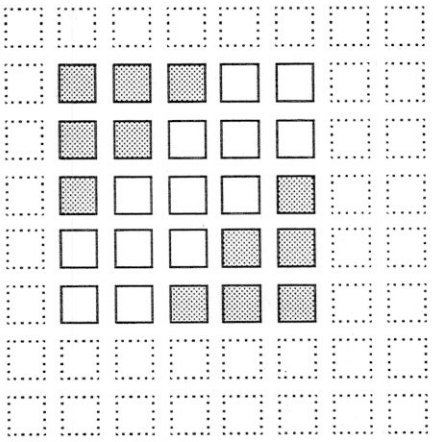
Figure 5.1b: The candidates for the *k*-th Smallest item ($7 \leq k \leq 12$)



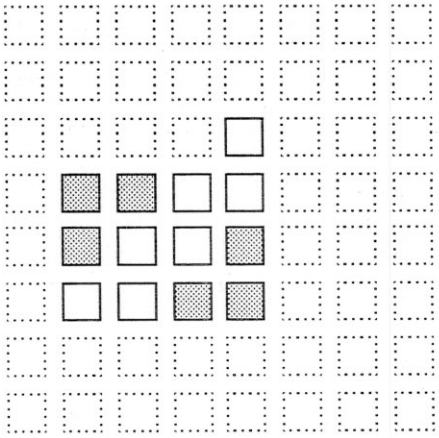
Step 1:



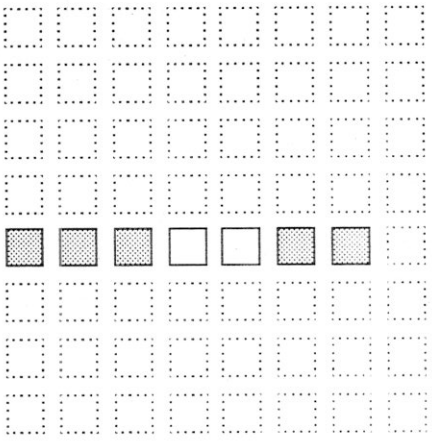
Step 2:



Step 3:



Step 4:



Step 5:

Figure 5.2: Median-Finding Algorithm (*N = 8x8)*