# Optimization of the Argus Omnidirectional Array with Software and Hardware Subarrays

Robert S. Dixon
Ohio State University
Columbus, Ohio, USA
Bob_Dixon@osu.edu

Presented at the 1kT (Square Kilometer Array) Workshop

December, 1997
Sydney, Australia

## Abstract

Argus is a wideband timed array that forms all possible beams simultaneously in real time, using digital beamforming. Since computing power is the ultimate limitation on the size and bandwidth of Argus, it is desirable to seek beamforming algorithms which minimize the amount of computation required. The common algorithm of creating a beam by delaying, weighting and summing the outputs of all N elements in an array requires computation proportional to N for each beam. For a close--spaced array the number of possible beams is about equal to the number of elements, so the total computation required is proportional to N squared. If a planar array is divided mathematically into identical subarrays, and the above algorithm used to first form beams for each subarray, and then to form the final beams, the total amount of computation required is reduced. The optimum size subarray is found to be the square root of the total array size, and with that choice the total computation is proportional to N to the 3/2 power. This process can be continued to any level of subarrays of subarrays, all requiring progressively less computation, until a subarray contains only 2 elements. At that point the computation required is proportional to N log N. and is equivalent to the Fast Fourier Transform. The disadvantage of software subarrays is the partial or total loss of the ability to tailor the beam shapes and sidelobes, and to create nulls in desired directions. Rather than simply subdividing an array mathematically, other types of optimization may be achieved by rearranging the elements in each subarray. Three--dimensional and volumetric subarray configurations provide more design freedom. One example is a spherical or hemispherical log -periodic structure which provides subarray beams that are frequency and look--direction independent.

## Brief Argus Background

Large omnidirectional timed array.

Elements have hemispherical coverage, pointed straight up.

Digital beamforming of ALL beams simultaneously.  No scanning.

No pointing.        Uses all available information.

Unlimited integration time.        RFI resistant.

Detects transient events.

Cost is limited by computing, which goes down with time.

Many Argus concepts are applicable to SKAI.

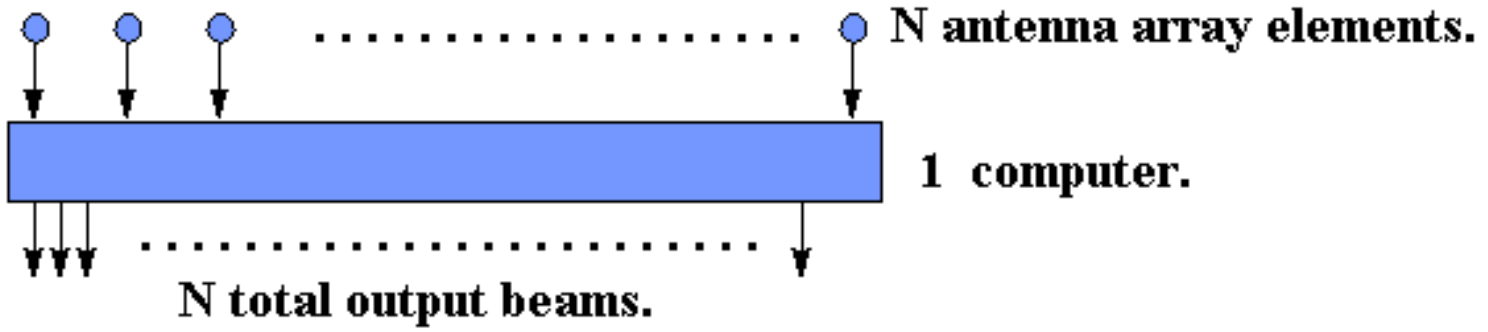
## Assumptions

1. The array is planar, with uniform half-wavelength spacings between elements.

2. The bandwidth is small, compared to the center frequency.

3. The beamforming cost is directly proportional to the total computing power required.
   (The proportionality constant is omitted.)

## Antenna Beamforming with Array Computing



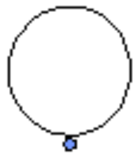
$C_T$ = Total computing power required

$$C_T = N * N = N^2$$

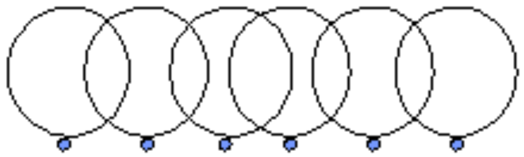
Bob Dixon, Ohio State University

## Antenna Beamforming using Subarrays

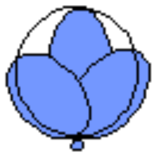
**Pattern of Individual Array Element**



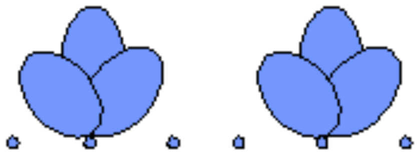
**Patterns of Individual Elements in a 6-Element Array**



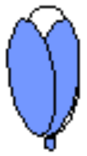
**Patterns of a 3-Element Subarray within an Individual Element Pattern**



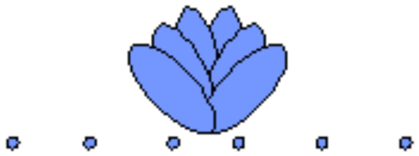
**Patterns of two Individual 3-Element Subarrays in a 6-Element Array**



**Patterns of two Arrayed 3-Element Subarrays within an Individual Subarray Pattern**

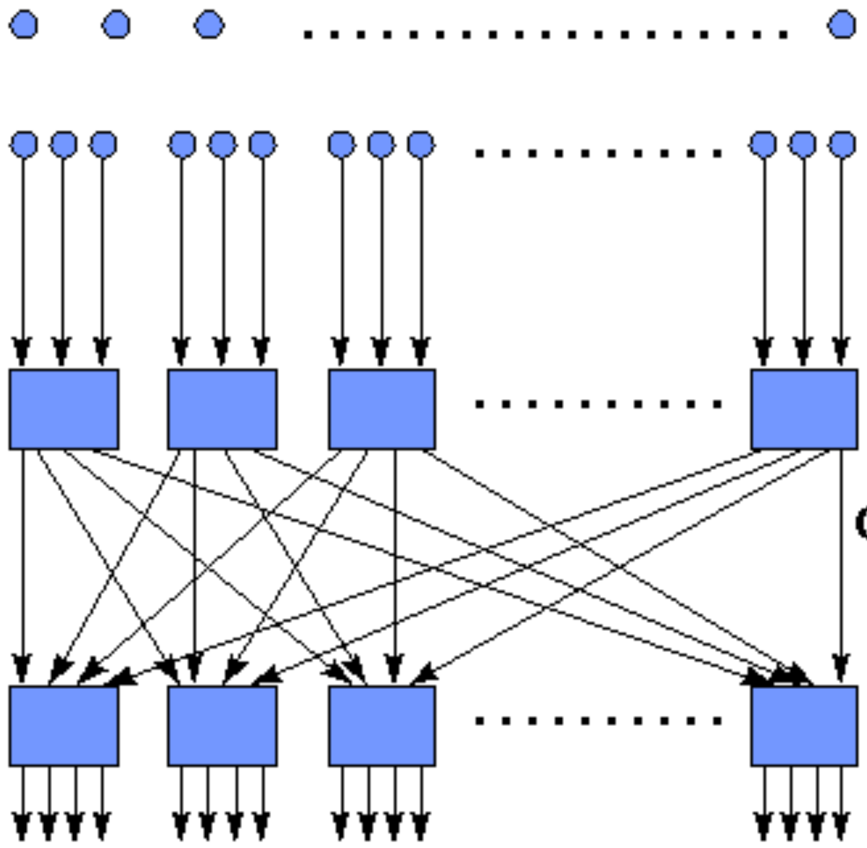


**Patterns of a 6-Element Array**



Bob Dixon, Ohio State University

# Antenna Beamforming with Subarray Computing



**N antenna array elements.**

**N elements in n subarrays.**
Each subarray has N/n elements.

**n subarray computers.**
Each computer has :
N/n elements input.
N/n subarray beams output.

$$C_S = (N/n)^2 * n = N^2/n$$

**N/n array computers.**
Each computer has :
n subarray beams input.
n array beams output.

**N total output beams.**

$$C_A = n^2 * (N/n) = nN$$

$$C_T = C_S + C_A = N^2/n + nN$$

**Bob Dixon, Ohio State University**

## What is the optimum subarray size ?

$$C_T = N^2 / n + nN$$

$$\frac{dC_T}{dn} = 0$$

$$n_{\text{optimum}} = N^{1/2}$$

$$C_{T\ \text{minimum}} = 2N^{3/2}$$

**Example: For a 1000 element array,**

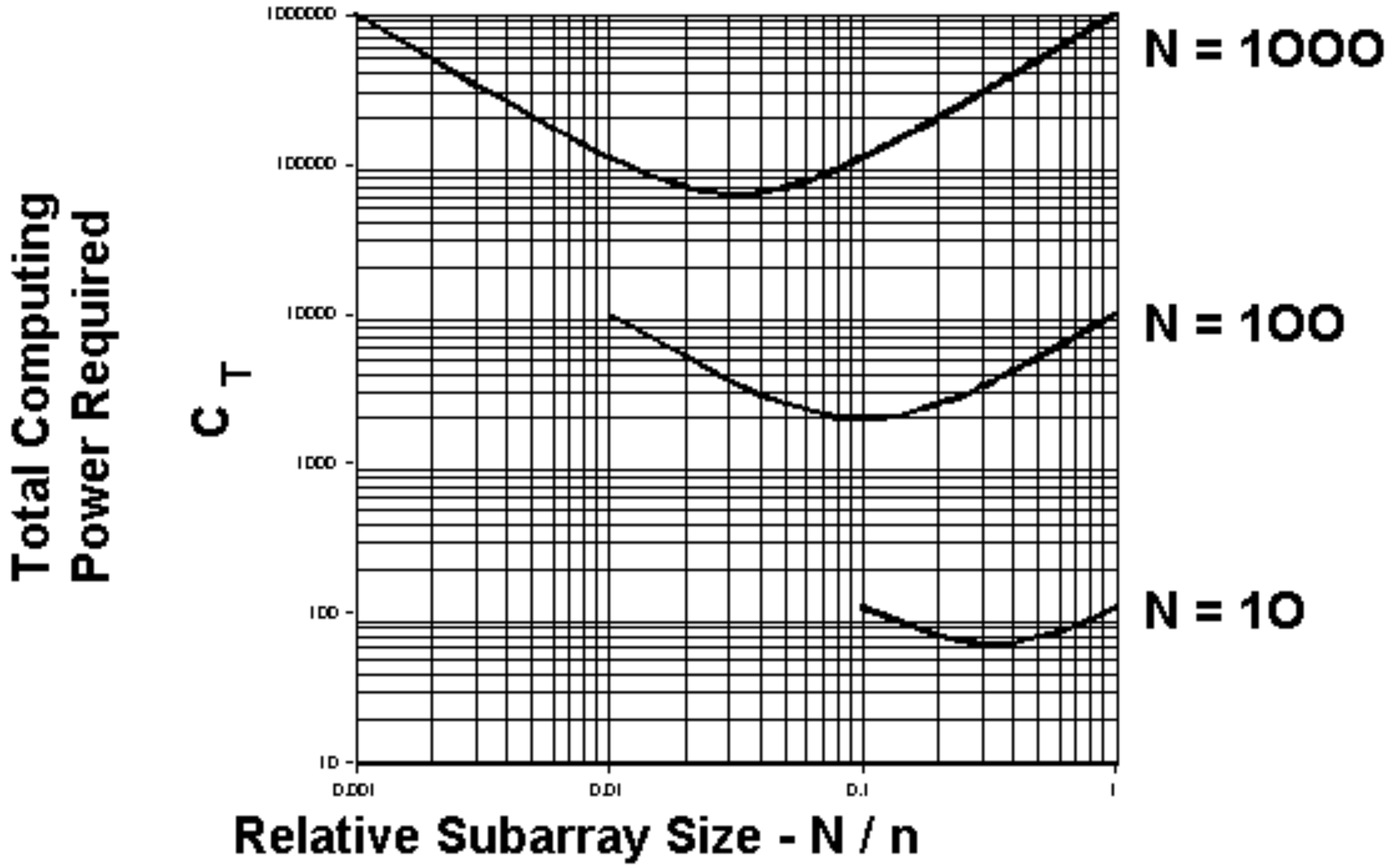
**$C_T$ = 1,000,000 (without subarrays)**

**$C_T$ = 63,000 (with subarrays)**

**The computing required is reduced by a factor of 16.**

**Bob Dixon, Ohio State Universit**

# Optimum Subarray Size



N = number of antenna elements

n = number of subarrays

## Consequences of Choosing the Optimum Subarray Size

1. The number of computers required is $2N^{1/2}$.

2. Each computer has a computing power of N.

3. All computers are identical.

4. All computers run the same software.

The subarray approach to beamforming requires less computing power, and is very suitable for parallel processing.

# Two levels of Subarrays

For a 3-level array, it is found that:

The optimum number of elements in each small subarray is $N^{1/3}$

The optimum number of small subarrays in each large subarray is $N^{1/3}$

The minimum computing required is $3\,N^{4/3}$

For the 1000 element example, $C_T = 30{,}000$

Going to 3 levels from 2 levels reduces the computing by another factor of 2.1 .

Bob Dixon, Ohio State University

## Generalization to any Number of Subarray Levels

| L | # computers | #computers/level | $C_1$ | $C_T$ |
|---|---|---|---|---|
| 1 | $1\ N^{0/1}$ | $N^{0/1}$ | $N^{2/1}$ | $1\ N^{2/1}$ |
| 2 | $2\ N^{1/2}$ | $N^{1/2}$ | $N^{2/2}$ | $2\ N^{3/2}$ |
| 3 | $3\ N^{2/3}$ | $N^{2/3}$ | $N^{2/3}$ | $3\ N^{4/3}$ |

We conjecture by extrapolation:

| | | | | |
|---|---|---|---|---|
| L | $L\ N^{(L-1)/L}$ | $N^{(L-1)/L}$ | $N^{2/L}$ | $L\ N^{(L+1)/L}$ |

Note that as the number of levels increases, the number of computers increases, but the power of each computer decreases.

## Is there an Optimum Number of Subarray Levels ?

$$C_T = L\,N^{(L+1)/L}$$

$$\frac{dC_T}{dL} = 0$$

$$L = \ln N$$

**The optimum number of subarray levels is the natural logarithm of the number of elements.**

## Consequences of Choosing the Optimum Number of Subarray Levels

1. The number of computers required is $\frac{N \ln N}{e}$

2. Each computer has a power of $C_1 = e^2$.

3. There are e elements in each smallest subarray.

4. All computers are identical and run the same software.

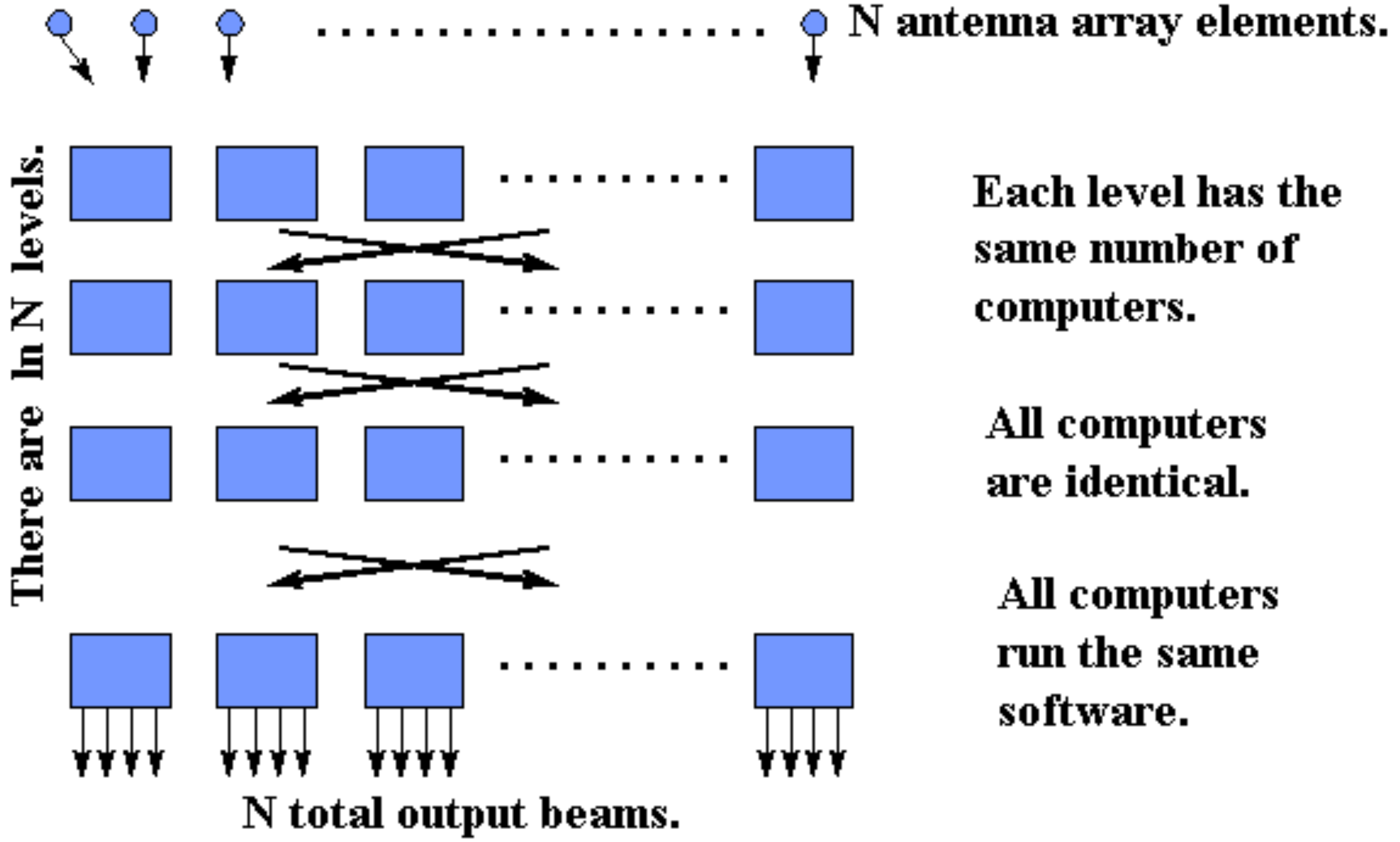
5. The total computing power required is $C_T = e\,N \ln N$ .

For the N=1000 example, $L_{optimum} = 7$

$$C_T = 18{,}800$$

(A further reduction of 1.6 from the N = 3 case.)

# Generalized Optimum Subarrays



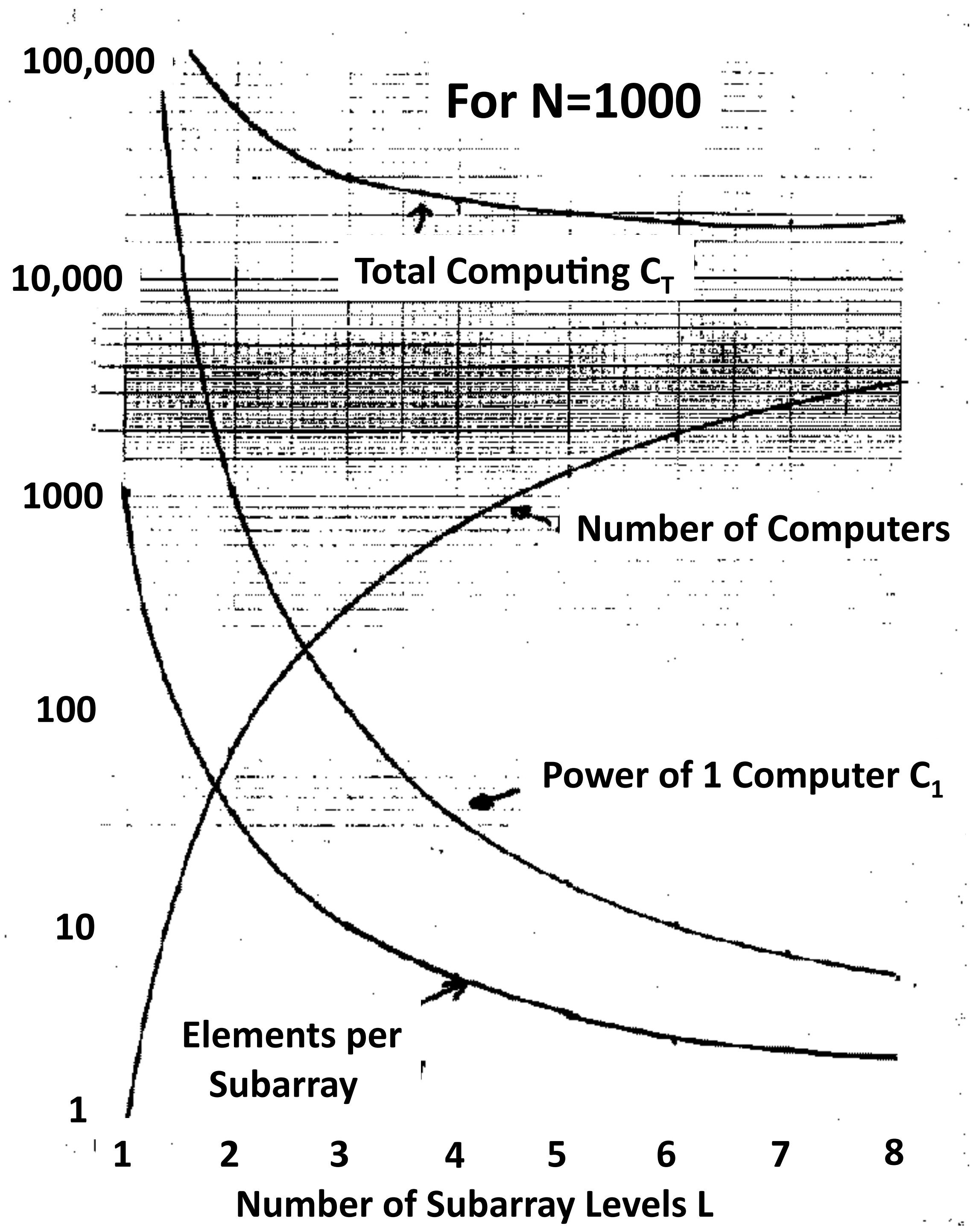
Each level has the same number of computers.

All computers are identical.

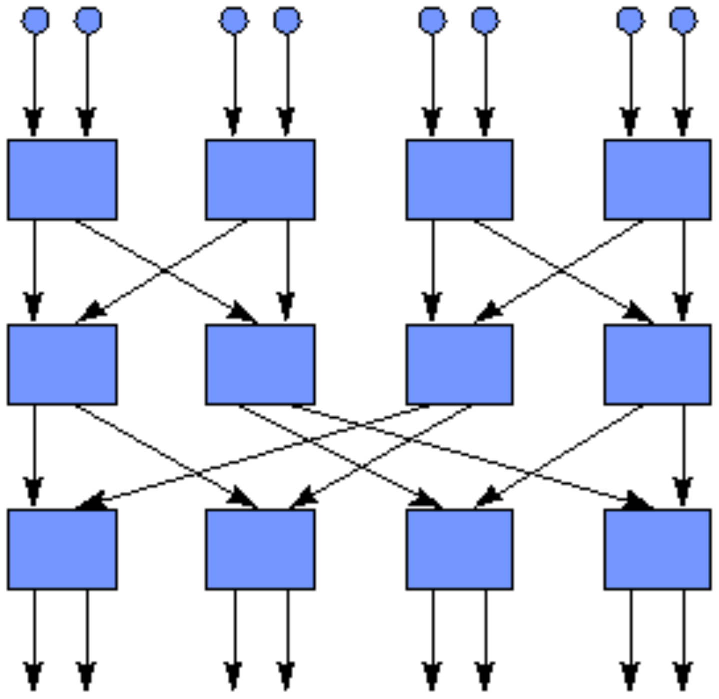
All computers run the same software.

Bob Dixon, Ohio State University

Results of choosing the optimum number of subarrays, for the example of a 1000 element array



# Binary Subarrays - The Extreme Smallest Case



**N total output beams.**

**$N = 2^L$ elements**

Each subarray has 2 elements.

**N/2 subarray computers, at each level.**

Each computer has :

2 elements or beams input.

2 beams output.

$$L_{optimum} = \log_2 N$$

$$\text{Number of Computers} = \frac{N \log_2 N}{2}$$

Power of each Computer = 4

$C_T = 2 N \log_2 N$ **For the N=1000 example, $C_T$= 20,000**

**Bob Dixon, Ohio State University**

## Restrictions:

The number of elements $N$ must be a perfect power.

## Unachievables:

The number of levels $L$ cannot be non-integer, such as $\ln N$.

## Tradeoffs:

1. The best control of sidelobes and nulls can only be obtained WITHOUT using subarrays. More subarray levels result in less control.
2. The lowest computing cost is obtained with the largest number of subarray levels.
3. There is a continuum of tradeoffs between antenna pattern control and computing cost.

# Hardware Optimizations of the Subarrays

Desireable Improvements over simple planar array:

- Frequency Independence
- Scan Angle Independence
- Larger element spacings and grating lobe control

1. James Breakall U.S. Patent
   - Log-periodic dipole arrays over planar ground plane
     - Frequency-independent planar array
     - Center-fire problem?
     - Scalability

2. Hemispherical ground plane geometry matches sky
   - Frequency-independent sectoral array
   - Sparse log-periodic dipoles
   - Scalable
   - Uses subset of elements

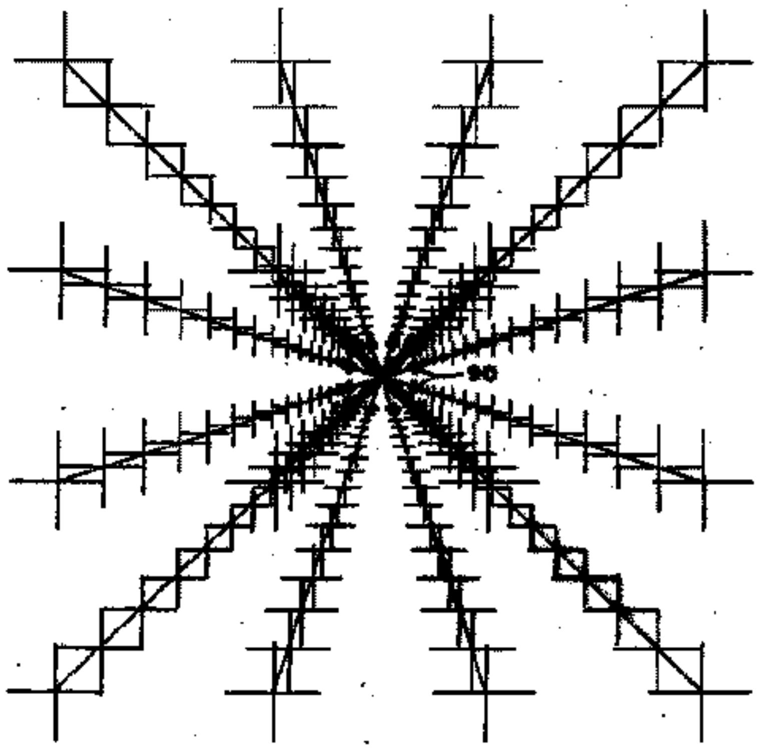


FIG. 7a

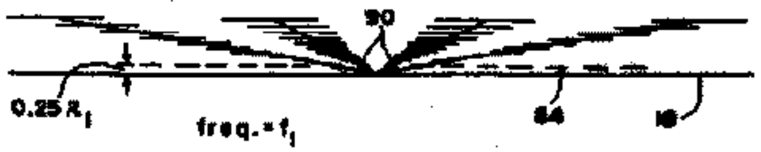


FIG. 7b