

Datalogi

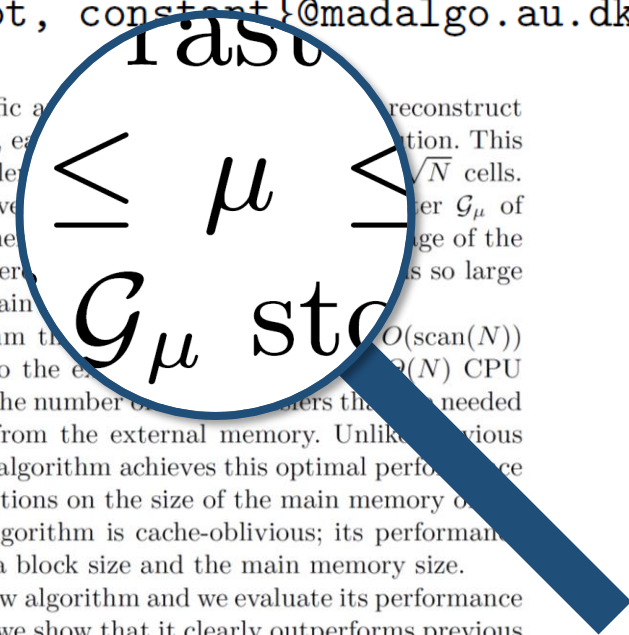
- et eksempel på datalogisk forskning

An optimal and practical cache-oblivious algorithm for computing multiresolution rasters

Lars Arge, Gerth Stølting Brodal, Jakob Truelsen, Constantinos Tsirogiannis

MADALGO*, Department of Computer Science, Aarhus University, Denmark
{large, gerth, jakobt, constant}@madalgo.au.dk

Abstract. In many scientific applications, one needs to reconstruct a raster dataset many times, e.g., for different resolutions. This leads to the following problem: Given a raster \mathcal{G} of \sqrt{N} cells. We want to compute for every resolution μ a raster \mathcal{G}_μ of $[\sqrt{N}/\mu] \times [\sqrt{N}/\mu]$ cells where each cell is the average of the values of $\mu \times \mu$ cells of \mathcal{G} . Here μ is so large that it does not fit in the main memory. We present a novel algorithm that uses only $O(\text{scan}(N))$ data block transfers from/to the external memory and $O(N)$ CPU operations; here $\text{scan}(N)$ is the number of operations needed to read the entire dataset from the external memory. Unlike previous results on this problem, our algorithm achieves this optimal performance without making any assumptions on the size of the main memory of the computer. Moreover, this algorithm is cache-oblivious; its performance does not depend on the data block size and the main memory size. We have implemented the new algorithm and we evaluate its performance on datasets of various sizes; we show that it clearly outperforms previous approaches on this problem. In this way, we provide solid evidence that non-trivial cache-oblivious algorithms can be implemented so that they perform efficiently in practice.



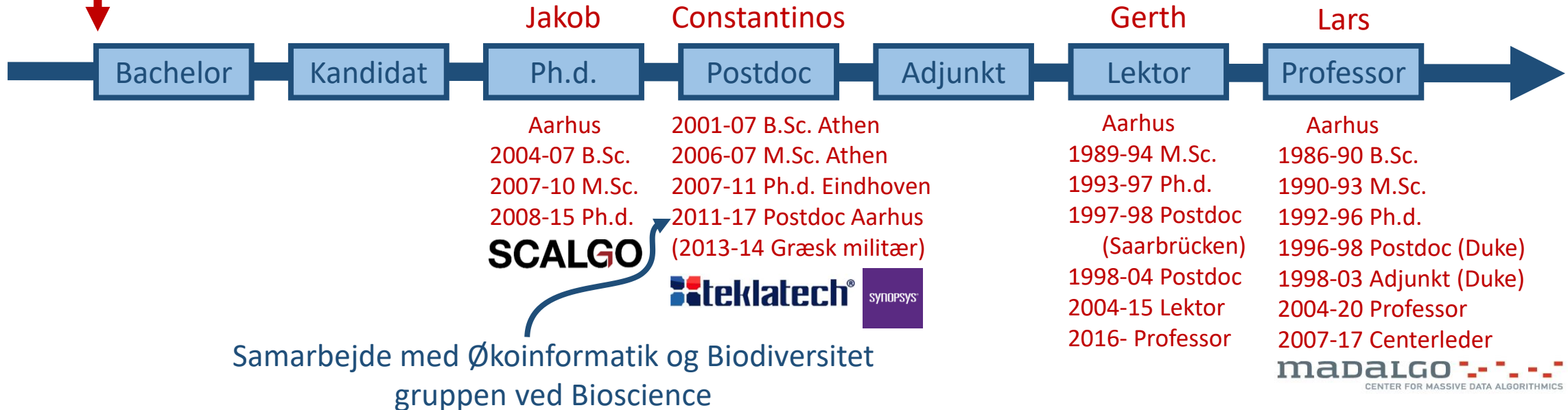
An optimal and practical cache-oblivious algorithm for computing multiresolution rasters

Lars Arge, Gerth Stølting Brodal, Jakob Truelsen, Constantinos Tsirogiannis

MADALGO*, Department of Computer Science, Aarhus University, Denmark

{large, gerth, jakobt, constant}@madalgo.au.dk

I er her



Samarbejde med Økoinformatik og Biodiversitet gruppen ved Bioscience

madalgo
CENTER FOR MASSIVE DATA ALGORITHMS

Eksempel på publikation (der anvender idéer fra ovenstående artikel):

Lars, Constantinos m.fl. "Natural surroundings in childhood are associated with lower schizophrenia rate", *Schizophrenia research*, 2020

Bachelor

Semester 1	Introduction to Programming	Algorithms and Data Structures	Introduction to Mathematics and Optimisation
Semester 2	Database Systems	Programming Languages	Computability and Logic
	Implementation and Applications of Databases		
Semester 3	Software Engineering and Architecture	Human-Computer Interaction	Introduction to Probability Theory and Statistics
Semester 4	Computer Architecture, Networks and Operating Systems	Experimental Systems Development	Numerical Linear Algebra
Semester 5	Compilation	Distributed Systems and Security	Machine Learning
Semester 6	Bachelor Project	Philosophy of Information Technology	Optimization

Kandidat

Semester 7	Computational Geometry: Theory and Experimentation	Program Analysis	Building the Internet of Things with P2P and Cloud Computing
Semester 8	Randomized Algorithms	Language-based Security	Augmented Reality
			Advanced Augmented Reality Project
Semester 9	Theory of Algorithms and Computational Complexity	Advanced Topics in Programming Language Theory	Deep Learning for Visual Recognition
Semester 10	Master Thesis		



Datalogi, obligatorisk




Matematik støttefag



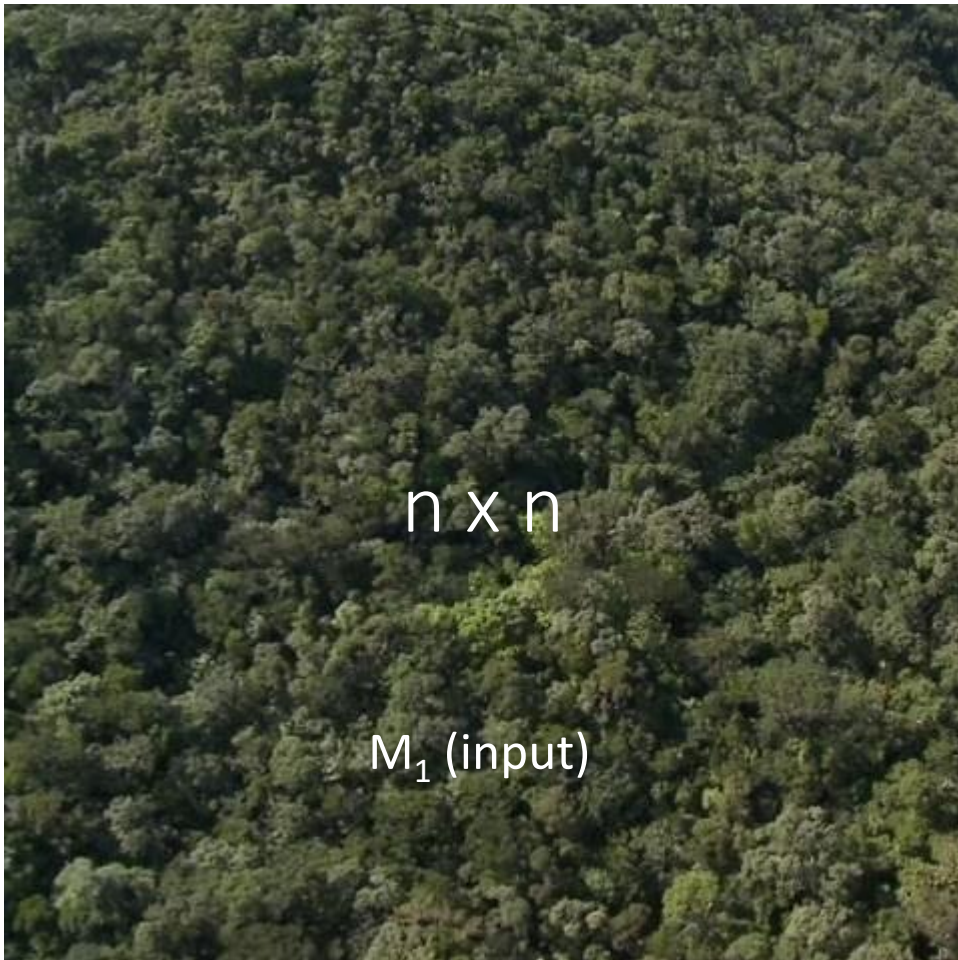
Valgfrie

An optimal and practical cache-oblivious algorithm for computing multiresolution **rasters**



 Rød: 64%
= Grøn: 70%
Blå: 44%

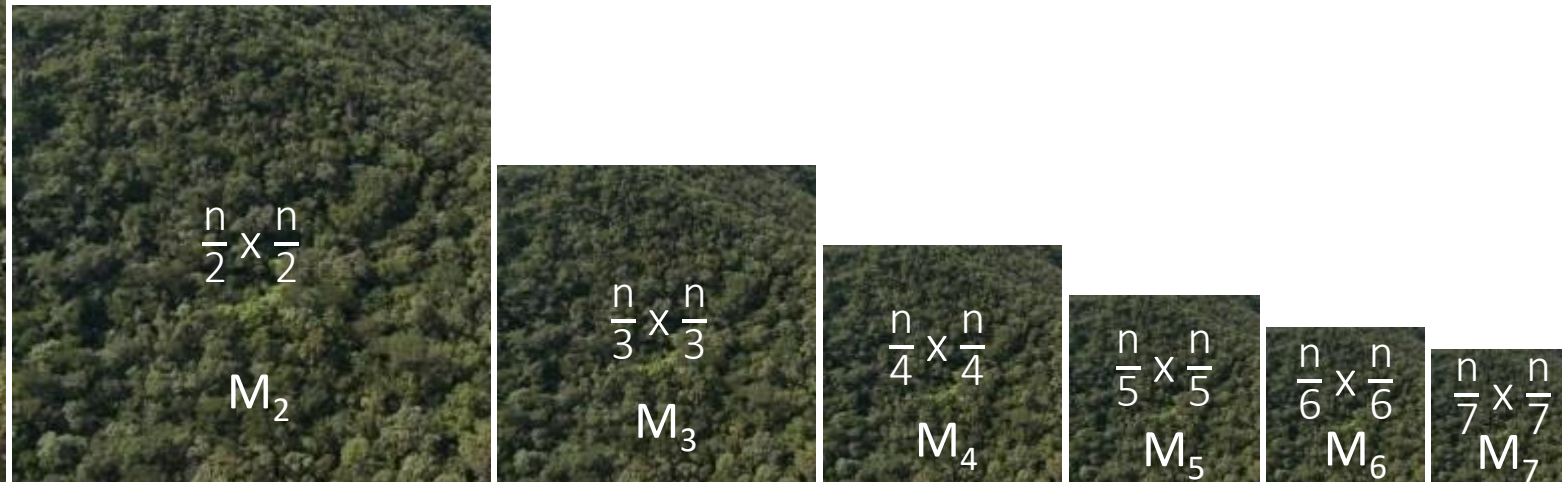
An optimal and practical cache-oblivious algorithm for computing **multiresolution rasters**



Input $n \times n$ billede M_1

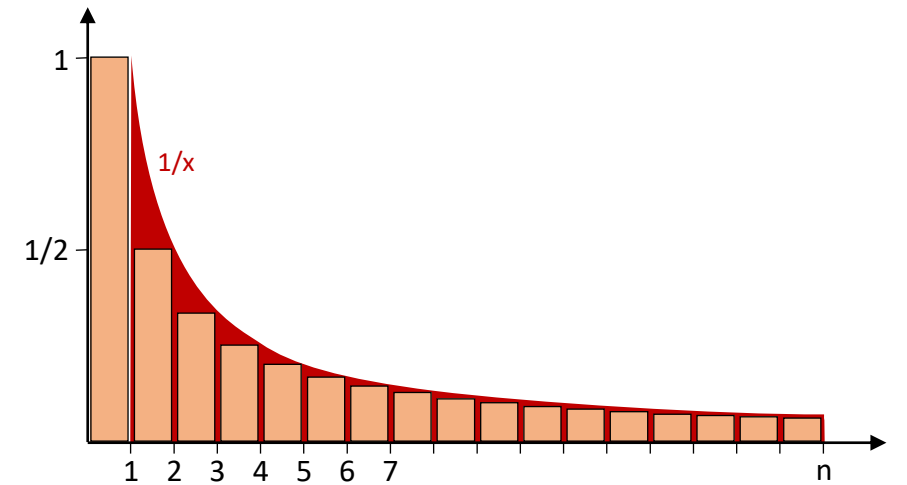
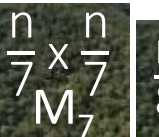
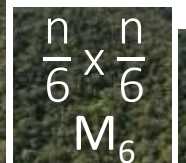
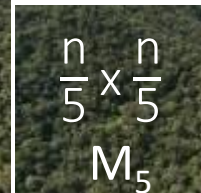
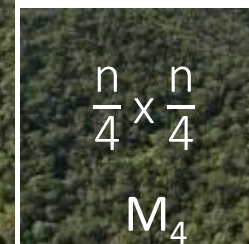
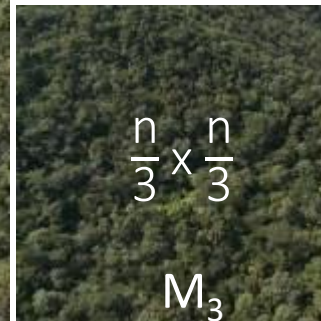
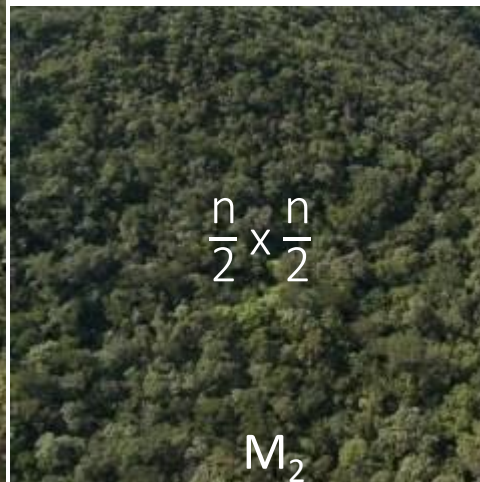
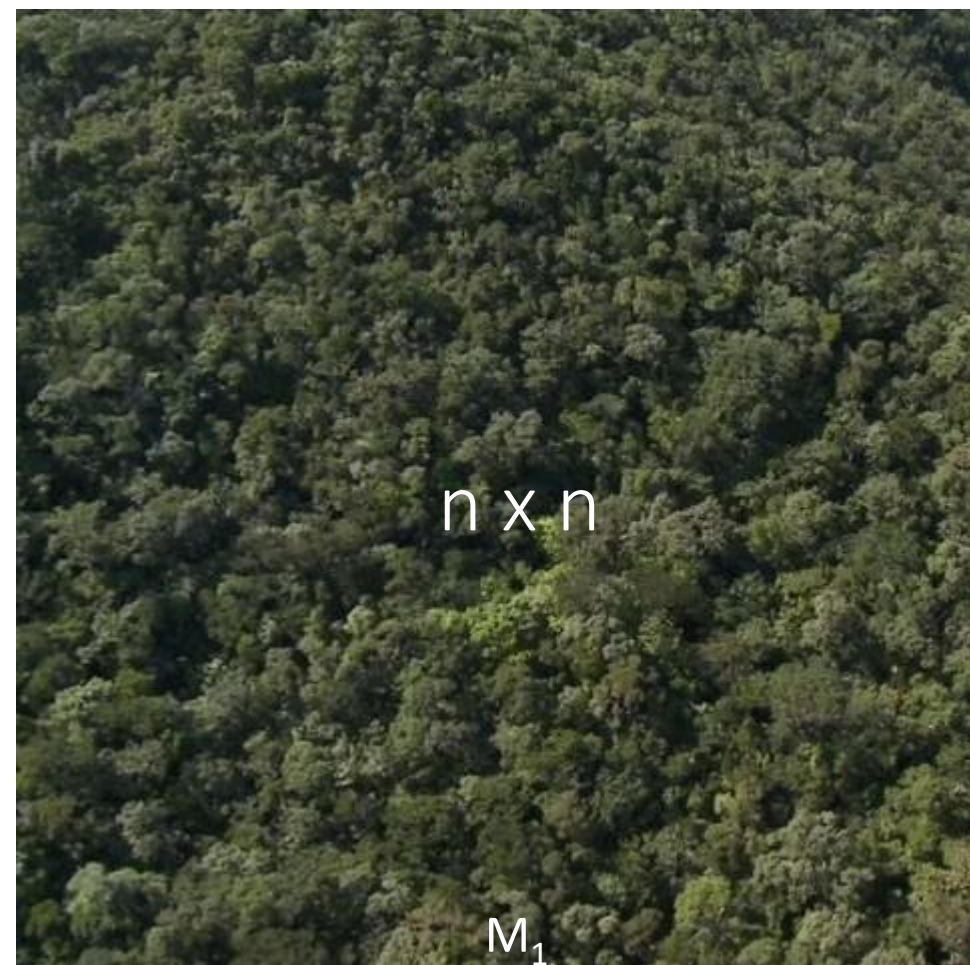
Output $M_2 M_3 \cdots M_d \cdots M_n$, hvor M_d størrelse $\frac{n}{d} \times \frac{n}{d}$

C.E. Woodcock and A.H. Strahler. The Factor of Scale in Remote Sensing.
Remote Sensing of Environments, 21:311-332, 1987

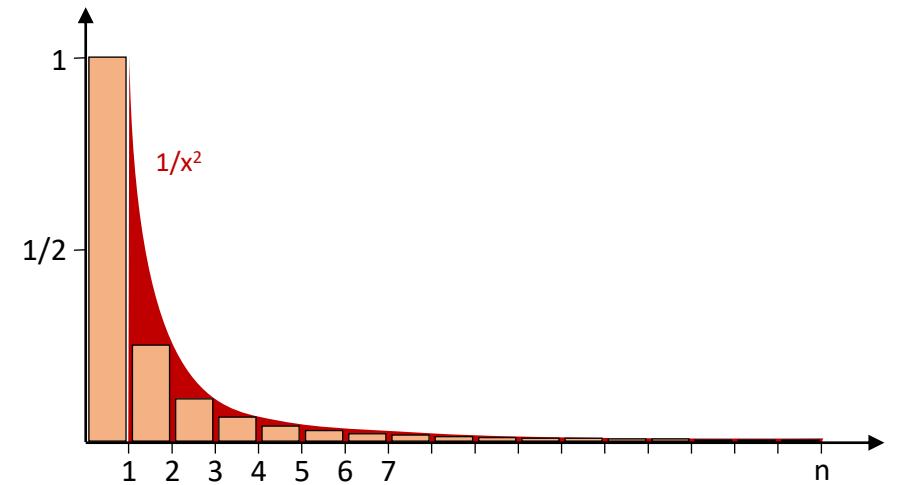
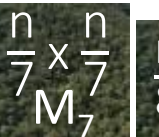
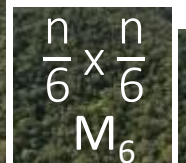
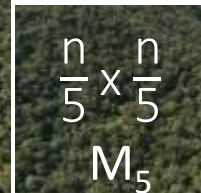
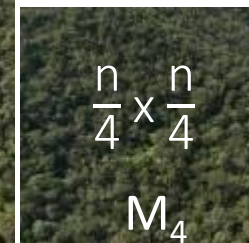
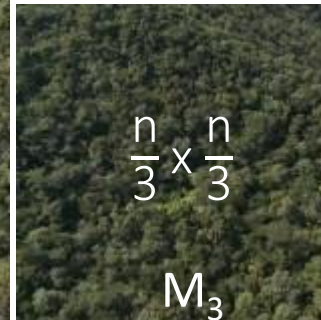
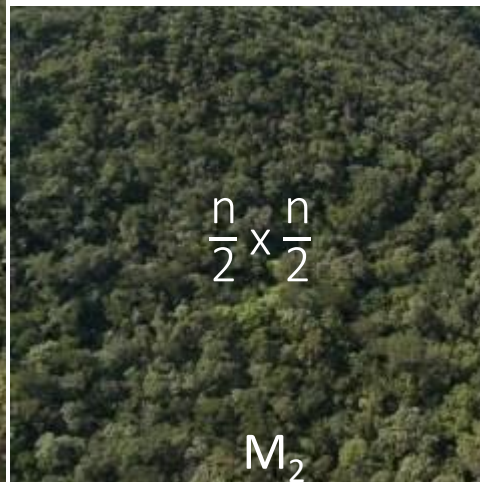
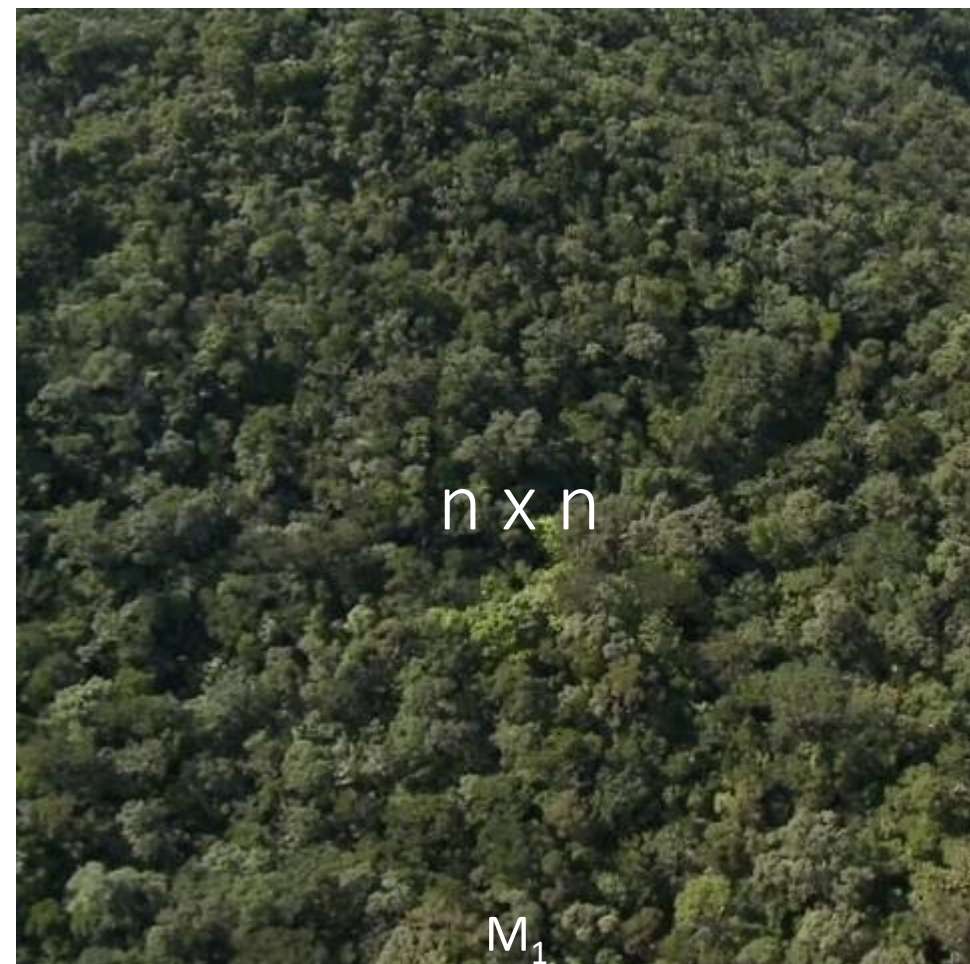


Samlet bredde = $\frac{n}{1} + \frac{n}{2} + \frac{n}{3} + \dots + \frac{n}{d} + \dots + \frac{n}{n-1} + \frac{n}{n} =$

$$\sum_{d=1}^n \frac{n}{d} = n \sum_{d=1}^n \frac{1}{d} \leq n \left(1 + \int_1^n \frac{1}{x} dx \right) = n(1 + [\ln(x)]_1^n) = n(1 + \ln(n))$$

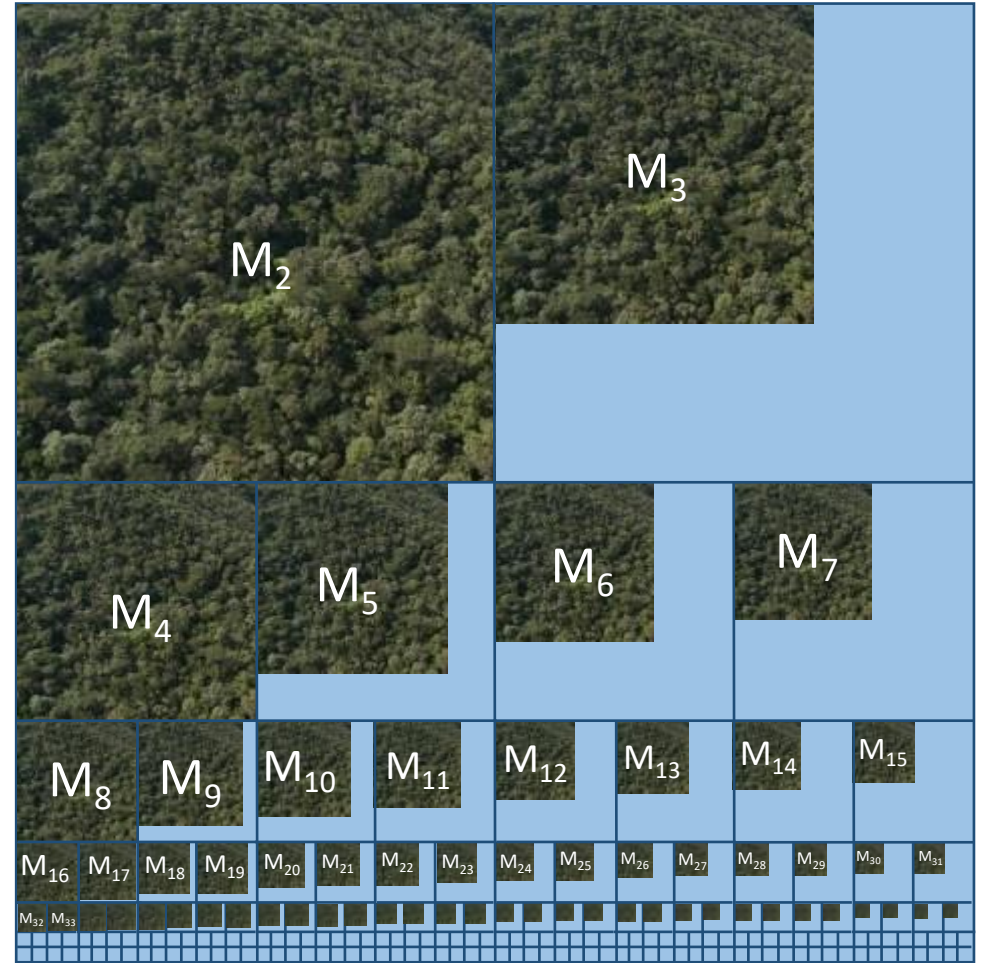
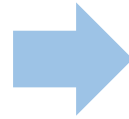


$$\begin{aligned} \text{Samlet størrelse} &= \binom{n}{1}^2 + \binom{n}{2}^2 + \binom{n}{3}^2 + \dots + \binom{n}{d}^2 + \dots + \binom{n}{n-1}^2 + \binom{n}{n}^2 \\ &= \sum_{d=1}^n \binom{n}{d}^2 = n^2 \sum_{d=1}^n \frac{1}{d^2} \leq n^2 \left(1 + \int_1^n \frac{1}{x^2} dx \right) = n^2 \left(1 + \left[-\frac{1}{x} \right]_1^n \right) \leq 2n^2 \end{aligned}$$



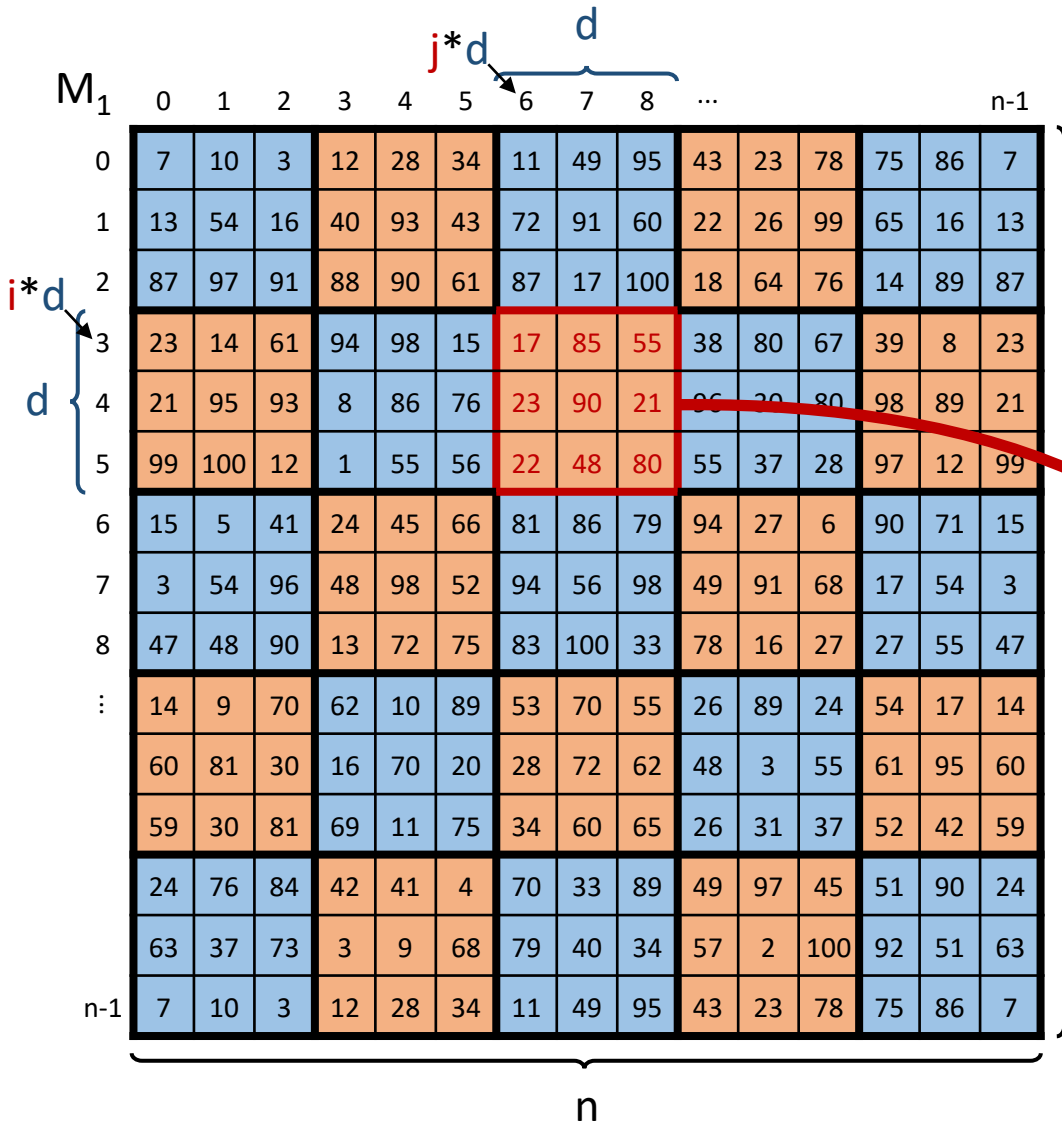


Input $n \times n$

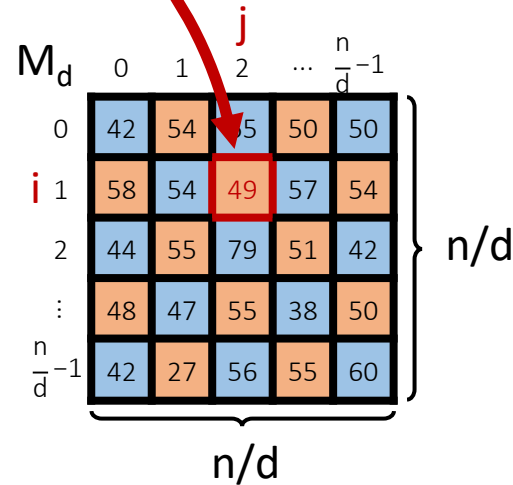


Output $n \times n$

An optimal and practical cache-oblivious algorithm for computing multiresolution rasters



Gennemsnit



```

Algoritme 1
for d = 2 to n
  for i = 0 to  $\lfloor n/d \rfloor - 1$ 
    for j = 0 to  $\lfloor n/d \rfloor - 1$ 
      sum = 0
      for r = 0 to d - 1
        for k = 0 to d - 1
          sum = sum +  $M_1[i*d + r, j*d + k]$ 
       $M_d[i, j] = \lfloor \text{sum} / d^2 \rfloor$ 
  
```

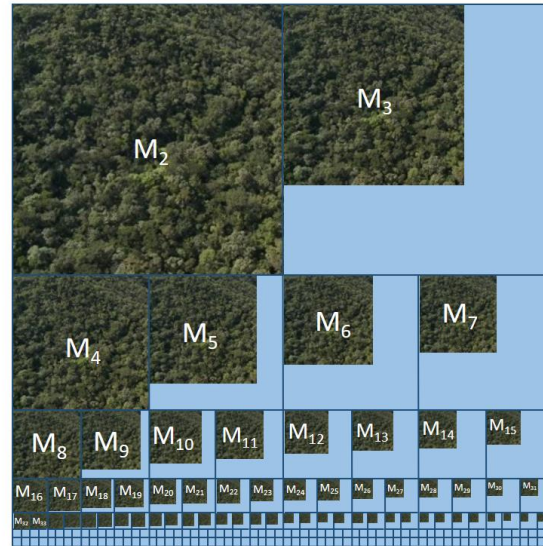
M_1 tilgås $\approx n^3$ gange

$$\left\lfloor \frac{17+85+55+23+90+21+22+48+80}{9} \right\rfloor = 49$$

An optimal and practical cache-oblivious algorithm for computing multiresolution rasters



Input $n \times n$



Output $n \times n$

Algoritme 1

```
for d = 2 to n
  for i = 0 to  $\lfloor n/d \rfloor - 1$ 
    for j = 0 to  $\lfloor n/d \rfloor - 1$ 
      sum = 0
      for r = 0 to d - 1
        for k = 0 to d - 1
          sum = sum +  $M_1[i*d + r, j*d + k]$ 
       $M_d[i, j] = \lfloor \text{sum}/d^2 \rfloor$ 
```

Tid $\approx n^3$

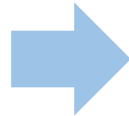
Input og output størrelse n^2 , men Algoritme 1 tager tid n^3 .

Kan man løse problemet hurtigere ?

M_1

	0	1	2	3	4	5	6	7	8	...	n-1				
0	7	10	3	12	28	34	11	49	95	43	23	78	75	86	7
1	13	54	16	40	93	43	72	91	60	22	26	99	65	16	13
2	87	97	91	88	90	61	87	17	100	18	64	76	14	89	87
3	23	14	61	94	98	15	17	85	55	38	80	67	39	8	23
4	21	95	93	8	86	76	23	90	21	96	30	80	98	89	21
5	99	100	12	1	55	56	22	48	80	55	37	28	97	12	99
6	15	5	41	24	45	66	81	86	79	94	27	6	90	B	15
7	3	54	96	48	98	52	94	56	98	49	91	68	17	54	3
8	47	48	90	13	72	75	83	100	33	78	16	27	27	55	47
⋮	14	9	70	62	10	89	53	70	55	26	89	24	54	17	14
	60	81	30	16	70	20	28	72	62	48	3	55	61	95	60
	59	30	81	69	11	75	34	60	65	26	31	37	52	42	59
	24	76	84	42	41	4	70	33	89	49	97	45	51	90	24
	63	37	73	3	9	68	79	40	34	57	2	100	92	51	63
n-1	7	10	3	12	28	34	11	49	95	43	23	78	75	86	7

sum



S

	-1	0	1	2	3	4	5	6	7	8	...	n-1				
-1	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	7	17	20	32	60	94	105	154	249	292	315	393	468	554	561
1	0	20	84	103	155	276	353	436	576	731	796	845	1022	1162	1264	1284
2	0	87	268	378	518	729	867	1037	1194	1449	1532	1645	1898	2052	2243	2350
3	0	130	305	476	710	1019	1372	1359	1601	1911	2032	2225	2545	2738	2937	3067
4	0	151	421	685	927	1322	1551	1761	2037	2424	2641	2864	3264	3555	3843	3994
5	0	250	620	896	1139	1589	1874	2106	2486	2897	3169	3429	3857	E	C	4795
6	0	265	640	957	1224	1719	2070	2383	2849	3339	3705	3992	4426	D	A	5540
7	0	268	697	1110	1425	2018	2421	2828	3350	3938	4353	4731	5233	5728	6153	6421
8	0	315	792	1295	1623	2288	2766	3256	3878	4499	4992	5386	5915	6437	6917	7232
⋮	0	329	815	1388	1778	2453	3020	3563	4255	4931	5450	5933	6486	7062	7559	7888
	0	389	956	1559	1965	2710	3297	3868	4627	5370	5937	6423	7031	7668	8260	8649
	0	448	1077	1727	2197	3028	3707	4368	5237	6000	6597	7123	7781	8474	9116	9519
	0	472	1111	1783	2317	3190	3927	4620	5507	6297	6917	7471	8161	8889	9568	9987
	0	535	1222	1917	2573	3508	4287	5022	5937	6747	7397	7991	8731	9511	10234	10803
n-1	0	542	1222	1917	2573	3508	4287	5022	5937	6747	7397	7991	8731	9511	10234	10803

Algorithme S

for $i = -1$ to $n - 1$
 $S[i, -1] = 0, S[-1, i] = 0$

for $i = 0$ to $n - 1$
 for $j = 0$ to $n - 1$
 $S[i, j] = M_1[i, j] + S[i-1, j] + S[i, j-1] - S[i-1, j-1]$

$Tid \approx n^2$

A
B
C
D
E

$17 + 85 + 55 + 23 + 90 + 21 + 22 + 48 + 80 = 2897 - 1874 - 1449 + 867$

An optimal and practical cache-oblivious algorithm for computing multiresolution rasters

S	-1	0	1	2	3	4	5	6	7	8	...	n-1				
-1	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	7	17	20	32	60	94	105	154	249	292	315	393	468	554	561
1	0	20	84	103	155	276	353	436	576	731	796	845	1022	1162	1264	1284
2	0	107	268	378	518	729	867	1037	1194	1449	1532	1645	1898	2052	2243	2350
3	0	130	305	476	710	1019	1172	1359	1601	1911	2032	2225	2545	2738	2937	3067
4	0	151	421	685	927	1322	1551	1761	2093	2424	2641	2864	3264	3555	3843	3994
5	0	250	620	896	1139	1589	1874	2106	2486	2897	3169	3429	3857	4245	4545	4795
6	0	265	640	957	1224	1719	2070	2383	2849	3339	3705	3992	4426	4904	5275	5540
7	0	268	697	1110	1425	2018	2421	2828	3350	3938	4353	4731	5233	5728	6153	6421
8	0	315	792	1295	1623	2288	2766	3256	3878	4499	4992	5386	5915	6437	6917	7232
⋮	0	329	815	1388	1778	2453	3020	3563	4255	4931	5450	5933	6486	7062	7559	7888
⋮	0	389	956	1559	1965	2710	3297	3868	4632	5370	5937	6423	7031	7668	8260	8649
⋮	0	448	1045	1729	2204	2960	3622	4227	5051	5854	6447	6964	7609	8298	8932	9380
⋮	0	472	1145	1913	2430	3227	3893	4568	5425	6317	6959	7573	8263	9003	9727	10199
⋮	0	535	1245	2086	2606	3412	4146	4900	5797	6723	7422	8038	8828	9660	10435	10970
n-1	0	542	1262	2106	2638	3472	4240	5005	5951	6972	7714	8353	9221	10128	10989	11531

Algorithm 2

for $d = 2$ to n

 for $i = 0$ to $\lfloor n/d \rfloor - 1$

 for $j = 0$ to $\lfloor n/d \rfloor - 1$

$r = i * d + d + 1, k = j * d + d + 1$

$M_d[i,j] = \lfloor (S[r,k] - S[r-d,k] - S[r,k-d] + S[r-d,k-d]) / d^2 \rfloor$

Tid $\approx n^2$

S tilgås $\leq 4n^2$ gange

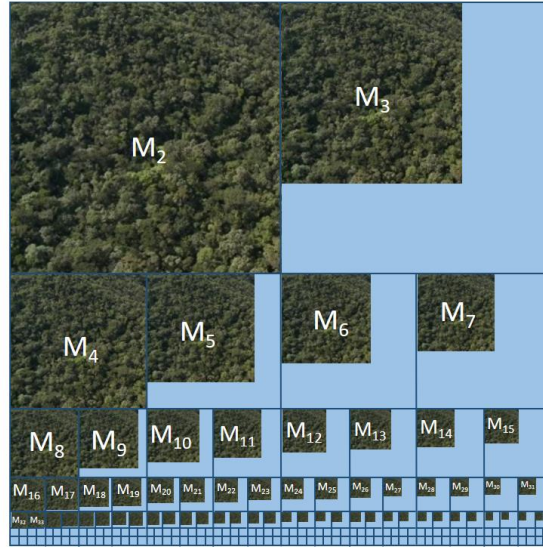
M_d	0	1	2	...	$\frac{n}{d}-1$
0	42	54	65	50	50
1	58	54	49	57	54
2	44	55	79	51	42
⋮	48	47	55	38	50
$\frac{n}{d}-1$	42	27	56	55	60

n/d

An optimal and practical cache-oblivious algorithm for computing multiresolution rasters



Input $n \times n$



Output $n \times n$

Algorithme 2

for $d = 2$ to n

 for $i = 0$ to $\lfloor n/d \rfloor - 1$

 for $j = 0$ to $\lfloor n/d \rfloor - 1$

$r = i * d + d + 1, k = j * d + d + 1$

$M_d[i,j] = \lfloor (S[r,k] - S[r-d,k] - S[r,k-d] + S[r-d,k-d]) / d^2 \rfloor$

$Tid \approx n^2$

Algorithme 2 tager optimal $tid \approx n^2$

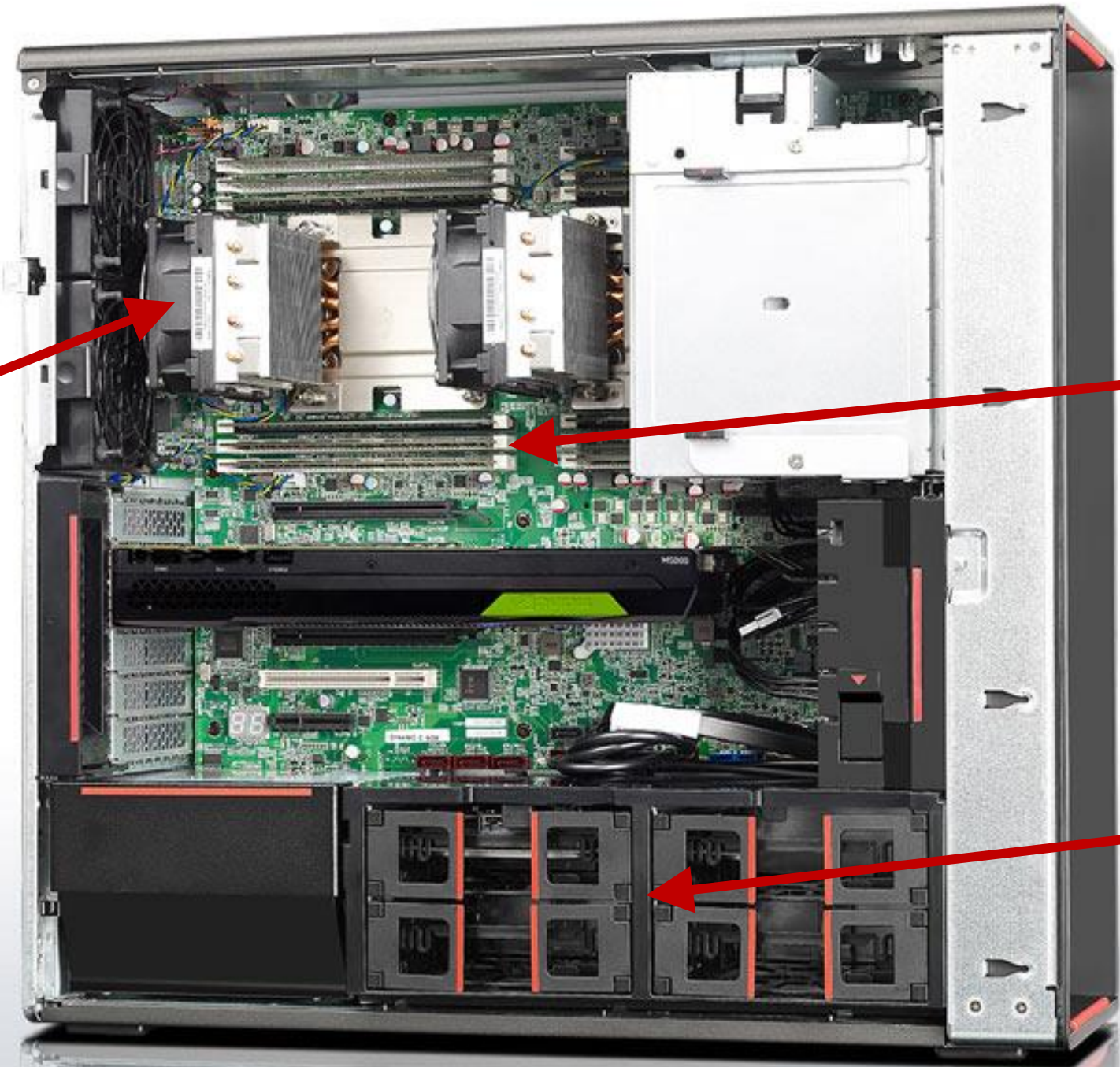
An optimal and practical **cache-oblivious** algorithm for computing multiresolution rasters

Algoritme 2 bruger den **fysiske hukommelse** uhensigtsmæssigt

Lenovo ThinkStation P710



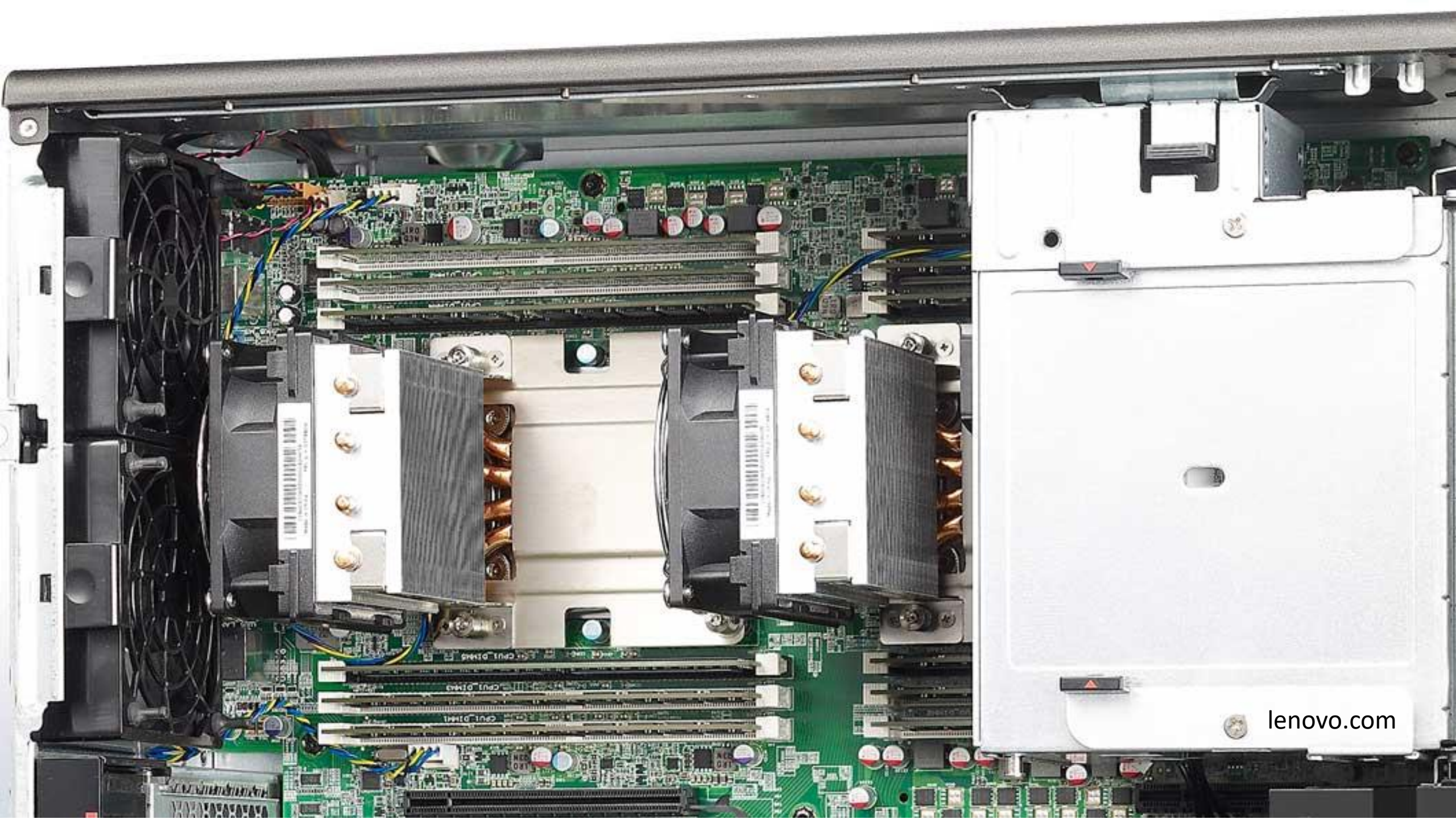
Lenovo ThinkStation P710



2 x CPU
2 x Intel Xeon
E5-2643 v4-processor

12 x hukommelse
12 x 32 GigaBytes
= 384×10^9 bytes

4 x harddiske
4 x 4 TeraBytes
= 16×10^{12} bytes

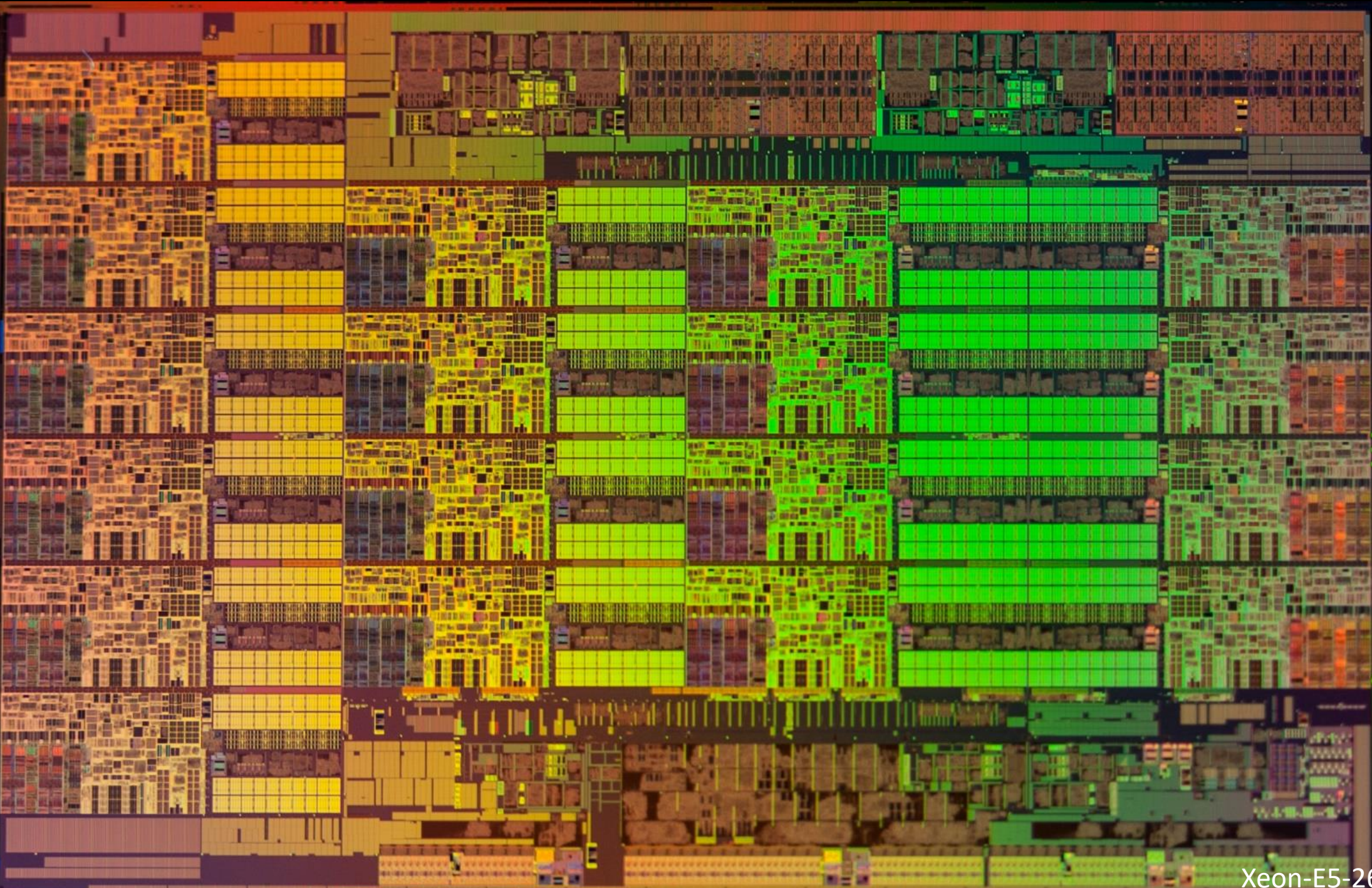


lenovo.com

Intel Xeon E5-2643 v4-processor

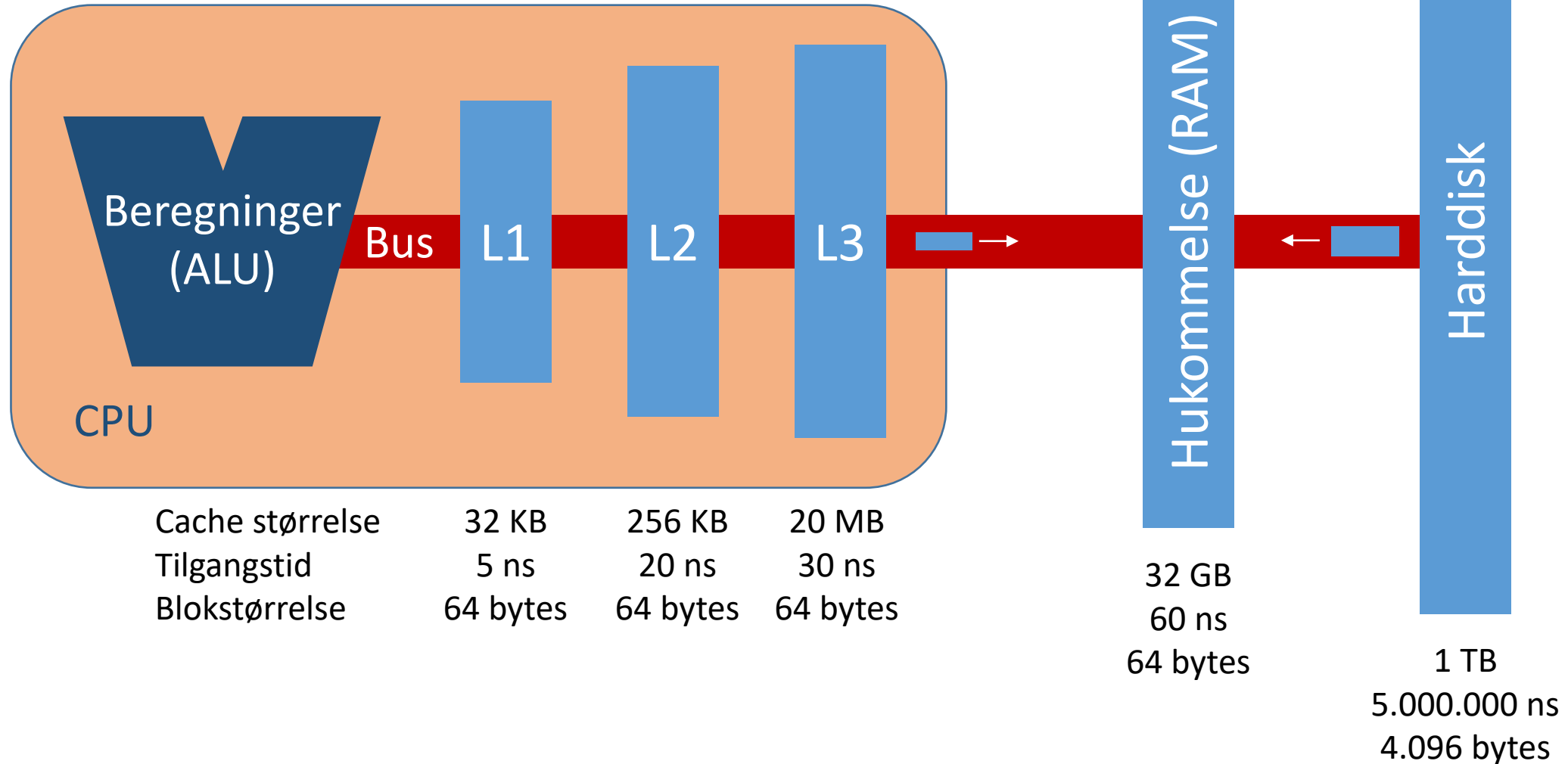


Transistors	3,200,000,000
The number of CPU cores	6
The number of threads	12
Level 1 instruction cache	6 x 32 KB 8-way set associative
Level 1 data cache	6 x 32 KB 8-way set associative
Level 2 cache	6 x 256 KB 8-way set associative
Level 3 cache	20 MB 20-way set associative shared
Cache line size	64 bytes
Data width	64 bit



Xeon-E5-2600-V3

Hukommelseshierarki



Algoritme 2 og Cache

Cache blokstørrelse = 4

S	-1	0	1	2	3	4	5	6	7	8	...	n-1				
-1	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	7	17	20	32	60	94	105	154	249	292	315	393	468	554	561
1	0	20	84	103	155	276	353	436	576	731	796	845	1022	1162	1264	1284
2	0	107	268	378	518	729	867	1037	1194	1449	1532	1645	1898	2052	2243	2350
3	0	130	305	476	710	1019	1172	1359	1601	1911	2032	2225	2545	2738	2937	3067
4	0	151	421	685	927	1322	1551	1761	2093	2424	2641	2864	3264	3555	3843	3994
5	0	250	620	896	1139	1589	1874	2106	2486	2897	3169	3429	3857	4245	4545	4795
6	0	265	640	957	1224	1719	2070	2383	2849	3339	3705	3992	4426	4904	5275	5540
7	0	268	697	1110	1425	2018	2421	2828	3350	3938	4353	4731	5233	5728	6153	6421
8	0	315	792	1295	1623	2288	2766	3256	3878	4499	4992	5386	5915	6437	6917	d
:	0	329	815	1388	1778	2453	3020	3563	4255	4931	5450	5933	6486	7062	7559	7888
	0	389	956	1559	1965	2710	3297	3868	4632	5370	5937	6423	7031	7668	8260	8649
	0	448	1045	1729	2204	2960	3622	4227	5051	5854	6447	6964	7609	8298	8932	9380
	0	472	1145	1913	2430	3227	3893	4568	5425	6317	6959	7573	8263	9003	9727	10199
	0	535	1245	2086	2606	3412	4146	4900	5797	6723	7422	8038	8828	9660	10435	10970
n-1	0	542	1262	2106	2638	3472	4240	5005	5951	6972	7714	8353	9221	10128	10989	11531

Algoritme 2

for d = 2 to n

 for i = 0 to $\lfloor n/d \rfloor - 1$

 for j = 0 to $\lfloor n/d \rfloor - 1$

 r = i * d, k = j * d

$M_d[i,j] = \lfloor (S[r,k] - S[r-d,k] - S[r,k-d] + S[r-d,k-d]) / d^2 \rfloor$

Tid $\approx n^2$

Hver tilgang til S kan være i en ny blok ☹️

M_6	0	1
0	52	55
1	49	56

An optimal and practical **cache-oblivious** **algorithm** for computing multiresolution rasters

Forskningsområde

- algoritmedesign
- udnytter at data kommer i blokke
- hardware parametrene må *ikke* indgå i algoritmen

Matteo Frigo, Charles E. Leiserson, Harald Prokop, Sridhar Ramachandran.
Cache-Oblivious Algorithms. *ACM Transactions on Algorithms*, 8(1), Article No. 4, 2012.

Primtal

Definition Et heltal p er et **primtal** hvis og kun hvis 1 og p går op i p

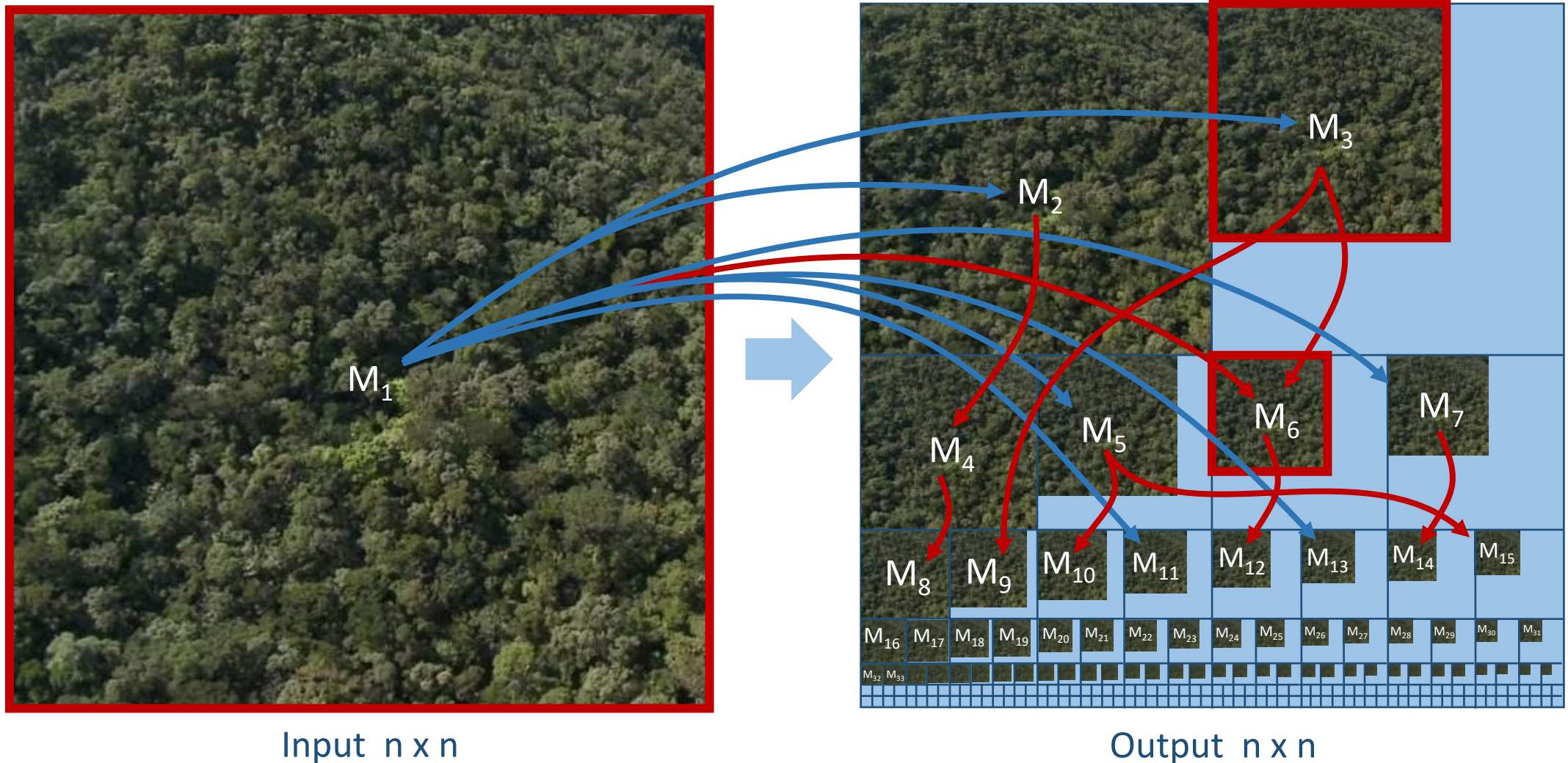
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67,
71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137,
139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, ...

Primtalssætningen Der er $\pi(n) < \frac{1.26 n}{\ln n}$ **primtal** mellem 2 og n

Primtalsfaktorisering

Definition Opskrivning af et heltal x som **produkt af primtal**

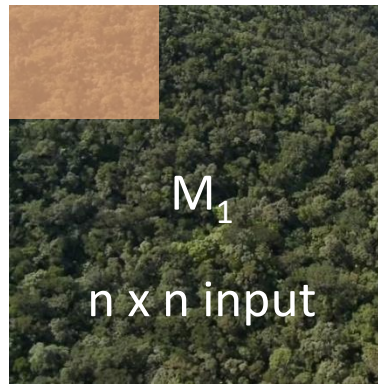
$$75460 = 2 * 2 * 5 * 7 * 7 * 7 * 11$$



Ide Konstruer M_d ud fra $M_{d/p}$, hvor p mindste primfaktor i d

Cache Oblivious Algoritme

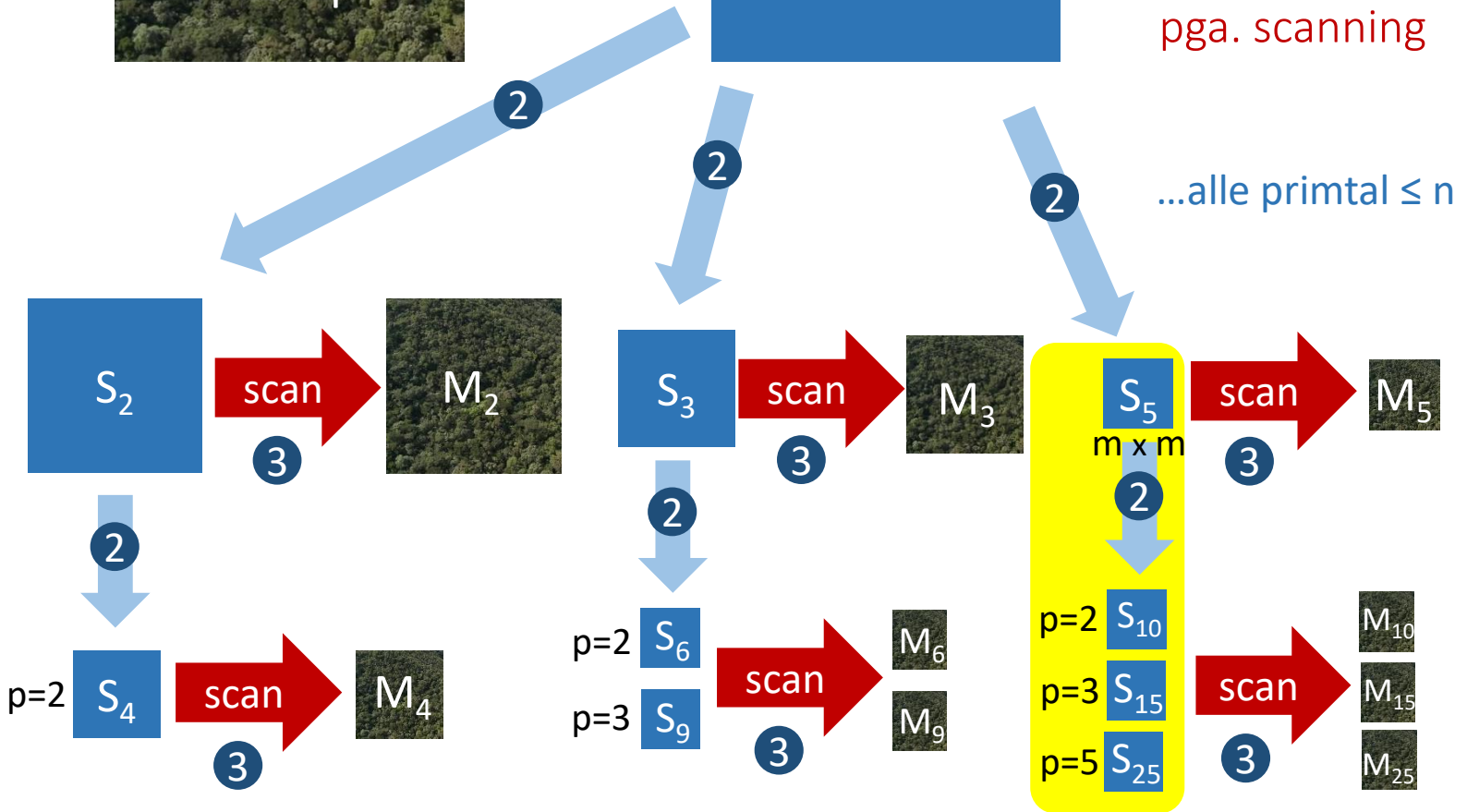
- Input n^2
- Output $\leq n^2$
- Sum $|S_i| \leq 2n^2$
- ① + ③ scanner $\leq 2n^2$
- ② gentagne scanninger og fjern alt som ikke skal bruges for større p



Algoritme S



Tid $\approx n^2$
og cache effektivt
pga. scanning



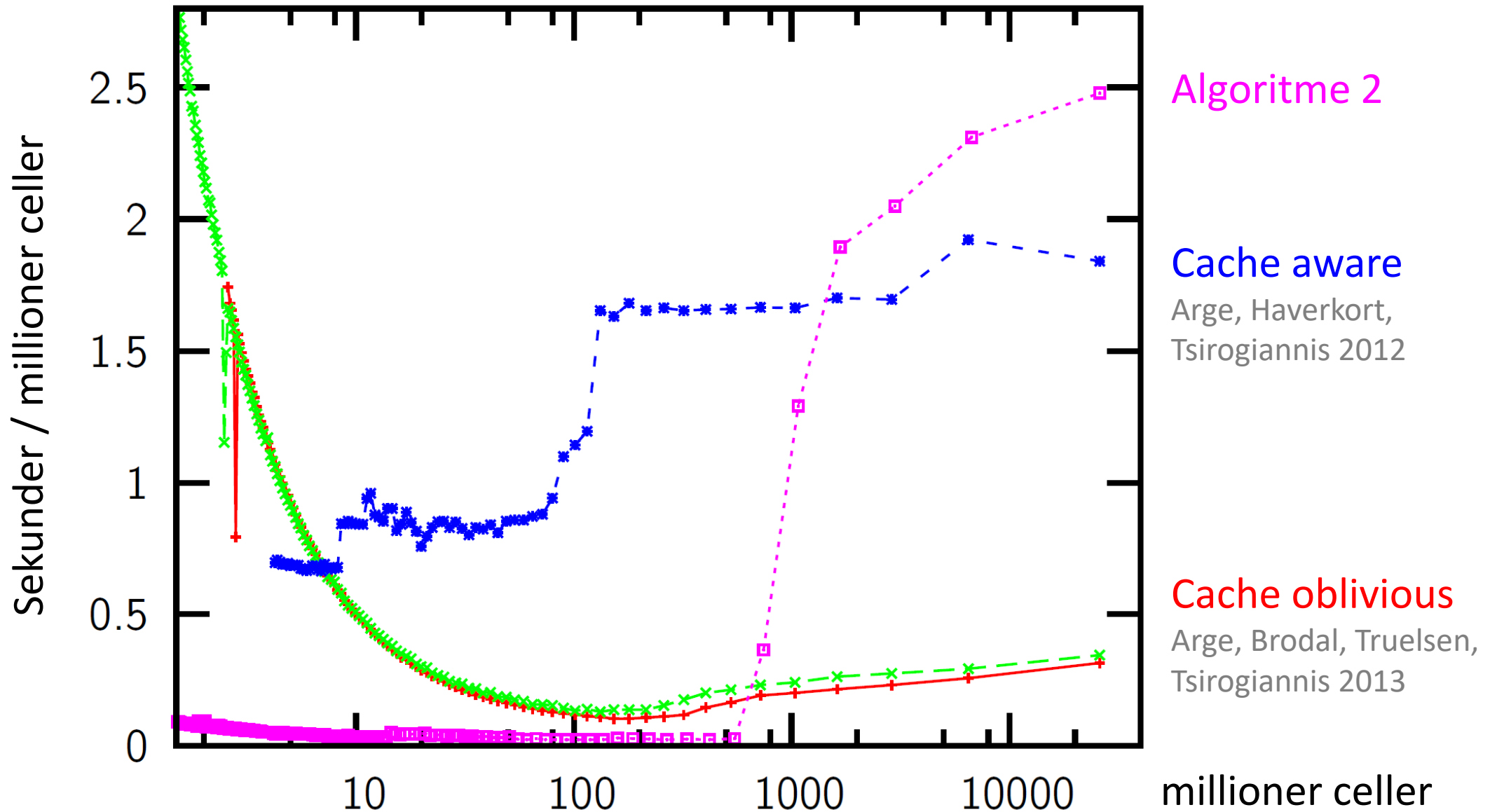
②

$$\sum_{\text{primtal } p} p \left(\frac{m}{p}\right)^2 \pi(p) \leq c \cdot m^2$$

$\pi(p)$ er antal primtal $\leq p$
Primtalssætningen

Eksempel ② for S_5 : Scan S_5 og uddrag S_{10} og fjern alt der ikke indgår i $S_{15} \cup S_{25}$.
Scan igen og uddrag S_{15} og fjern alt der ikke indgår i S_{25} .

Ekperimentelle Resultater



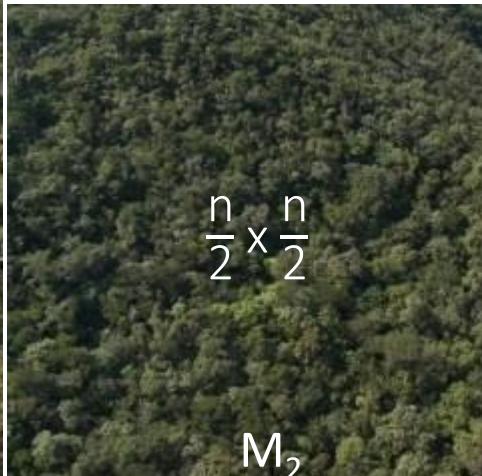
Opsummering

- Eksempel på datalogisk forskning (med biologisk anvendelse)
- Kombination af teori og praksis
- Matematik et centralt værktøj



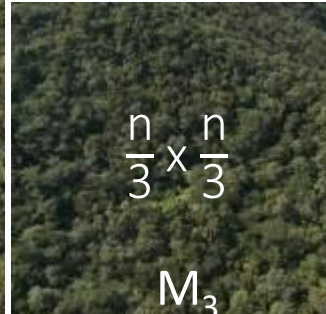
$n \times n$

M_1



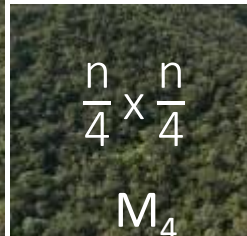
$\frac{n}{2} \times \frac{n}{2}$

M_2




$\frac{n}{3} \times \frac{n}{3}$

M_3



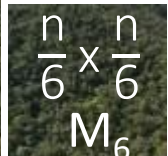
$\frac{n}{4} \times \frac{n}{4}$

M_4



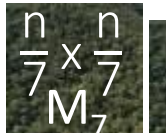
$\frac{n}{5} \times \frac{n}{5}$

M_5



$\frac{n}{6} \times \frac{n}{6}$

M_6



$\frac{n}{7} \times \frac{n}{7}$

M_7

Bachelor

Semester 1	Introduction to Programming	Algorithms and Data Structures	Introduction to Mathematics and Optimisation
Semester 2	Database Systems	Programming Languages	Computability and Logic
	Implementation and Applications of Databases		
Semester 3	Software Engineering and Architecture	Human-Computer Interaction	Introduction to Probability Theory and Statistics
Semester 4	Computer Architecture, Networks and Operating Systems	Experimental Systems Development	Numerical Linear Algebra
Semester 5	Compilation	Distributed Systems and Security	Machine Learning
Semester 6	Bachelor Project	Philosophy of Information Technology	Optimization

Kandidat

Semester 7	Computational Geometry: Theory and Experimentation	Program Analysis	Building the Internet of Things with P2P and Cloud Computing
Semester 8	Randomized Algorithms	Language-based Security	Augmented Reality
			Advanced Augmented Reality Project
Semester 9	Theory of Algorithms and Computational Complexity	Advanced Topics in Programming Language Theory	Deep Learning for Visual Recognition
Semester 10	Master Thesis		



Datalogi, obligatorisk



Matematik støttefag



Valgfrie

Velkommen til Datalogi