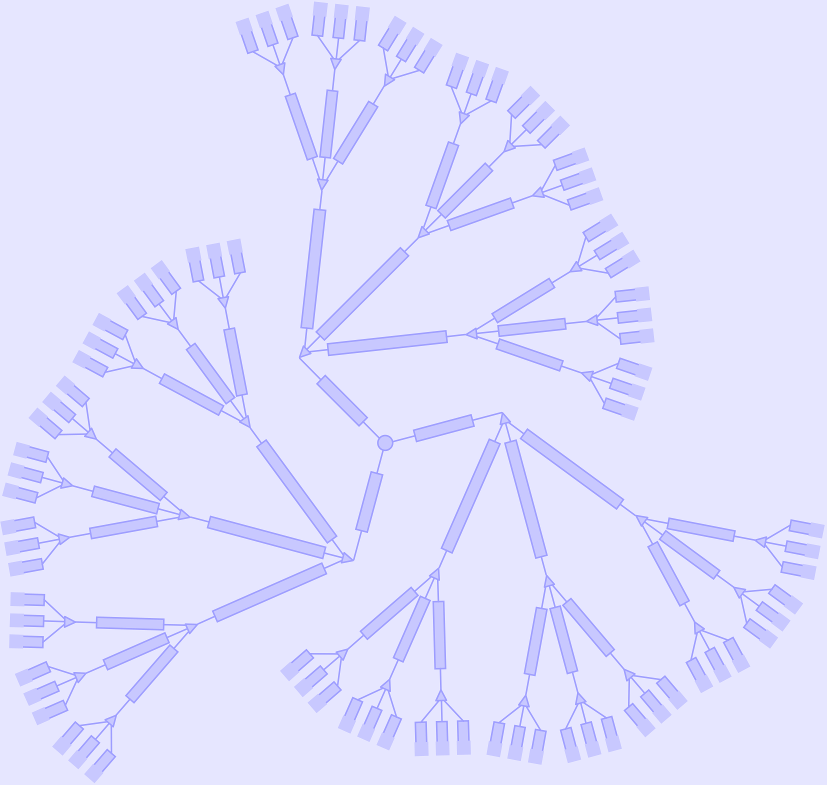
# Algorithms and Data Structures for Hierarchical Memory



**Gerth Stølting Brodal**
University of Aarhus

BRICS, University of Aarhus, May 26, 2004

# BRICS people

| | |
|---|---|
| Lars Arge, Professor | Joining September 1, 2004, SNF Rømer |
| Gerth Stølting Brodal, Lektor | Carlsberg |
| Rolf Fagerberg, Lektor | Since March 1, 2004, Odense |
| Herman J. Haverkort, Post. Doc. | Joining October 1, 2004, SNF |
| Gabriel Moruz, Ph.D. student | |

Former BRICS students with hierarchical memory research

| | |
|---|---|
| Lars Arge | Ph.D. 1996, Duke University |
| Gerth Stølting Brodal | Ph.D. 1997, University of Aarhus |
| Jakob Pagter | Ph.D. 2001, University of Aarhus |
| Riko Jacob | Ph.D. 2002, ETH Zürich |

# Typical workstations...

# Typical workstations...

families of

# Customizing a Dell 650 May 26, 2004

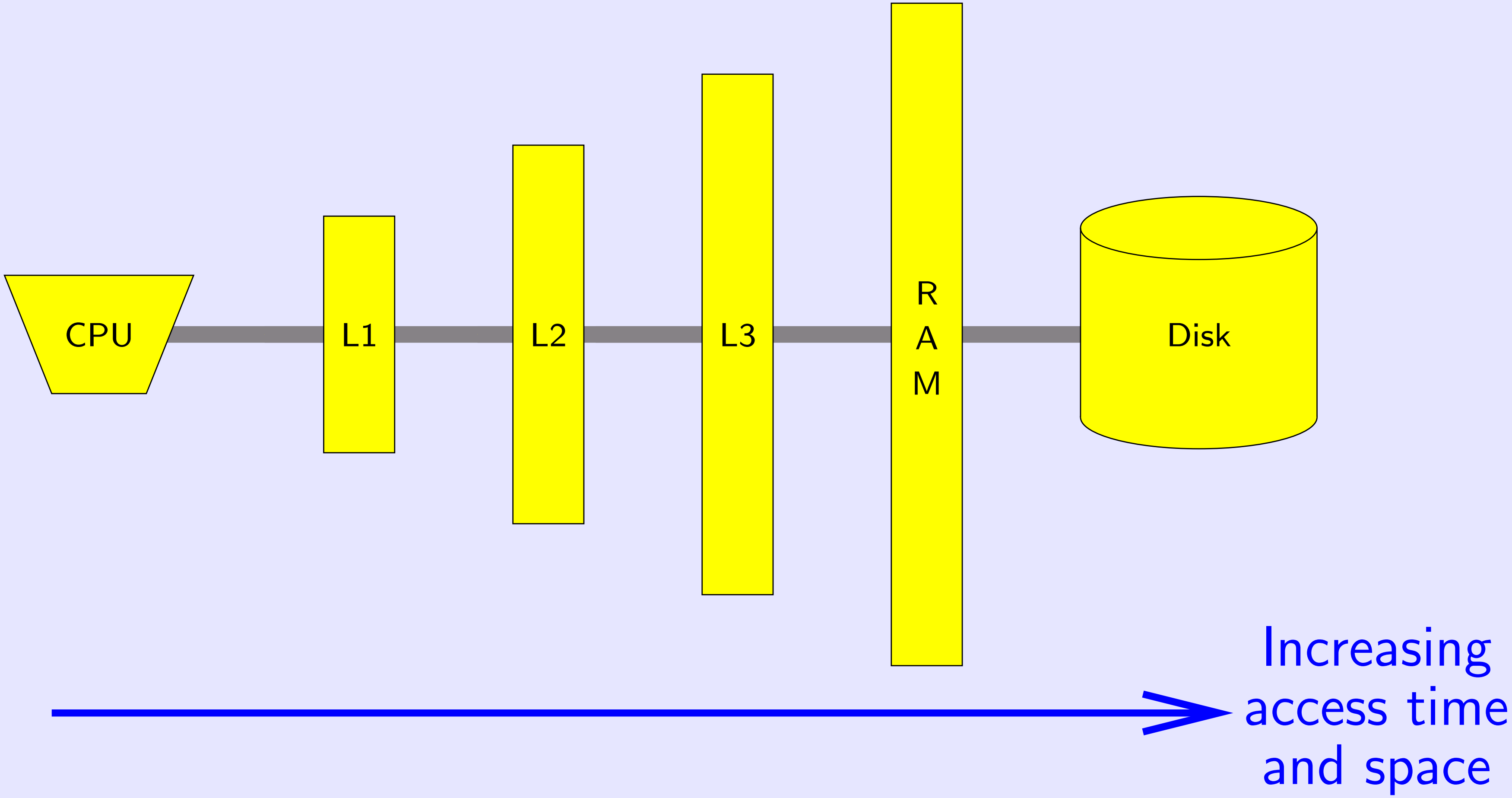
www.dell.dk

| | |
|---|---|
| Processor speed | 2.4 – 3.2 GHz |
| L3 cache size | 0.5 – 2 MB |
| Memory | 1/4 – 4 GB |
| Hard Disk | 36 GB – 146 GB |
| | 7.200 – 15.000 RPM |
| CD/DVD | 8 – 48x |

www.intel.com

| | |
|---|---|
| L2 cache size | 256 – 512 KB |
| L2 cache line size | 128 Bytes |
| L1 cache line size | 64 Bytes |
| L1 cache size | 16 KB |

# Hierarchical Memory Basics



- Data moved between adjacent memory levels in blocks

# More Hardware Specifications...

| | *Pentium 4* | *Pentium III* | *MIPS 10000* | *AMD Athlon* | *Itanium 2* |
|---|---|---|---|---|---|
| Architecture type | Modern CISC | Classic CISC | RISC | Modern CISC | EPIC |
| Operation system | Linux v. 2.4.18 | Linux v. 2.4.18 | IRIX v. 6.5 | Linux 2.4.18 | Linux 2.4.18 |
| Clock rate | 2400MHz | 800MHz | 175MHz | 1333 MHz | 1137 MHz |
| Address space | 32 bit | 32 bit | 64 bit | 32 bit | 64 bit |
| Pipeline stages | 20 | 12 | 6 | 10 | 8 |
| L1 data cache size | 8 KB | 16 KB | 32 KB | 128 KB | 32 KB |
| L1 line size | 128 B | 32 B | 32 B | 64 B | 64 B |
| L1 associativity | 4-way | 4-way | 2-way | 2-way | 4-way |
| L2 cache size | 512 KB | 256 KB | 1024 KB | 256 KB | 256 KB |
| L2 line size | 128 B | 32 B | 32 B | 64 B | 128 B |
| L2 associativity | 8-way | 4-way | 2-way | 8-way | 8-way |
| TLB entries | 128 | 64 | 64 | 40 | 128 |
| TLB associativity | full | 4-way | 64-way | 4-way | full |
| TLB miss handling | hardware | hardware | software | hardware | ? |
| RAM size | 512 MB | 256 MB | 128 MB | 512 MB | 3072 MB |

# Motivation

- Memory hierarchy has become a fact of life
- Accessing non-local storage may take a very long time
- Good locality is important to achieving high performance
- Handling massive data requires optimal memory usage

| | Latency | Relative to CPU |
|---|---:|---:|
| Register | 0.5 ns | 1 |
| L1 cache | 0.5 ns | 1-2 |
| L2 cache | 3 ns | 2-7 |
| DRAM | 150 ns | 80-200 |
| TLB | 500+ ns | 200-2000 |
| Disk | 10 ms | $10^7$ |

Increasing

# Algorithmic Question

- Modern hardware is not uniform — *many* different parameters
  - Number of caches
  - Cache sizes
  - Cache line/disk block sizes
  - Cache associativity
  - Cache replacement strategy
  - CPU/BUS/memory speed

# Algorithmic Question

- Modern hardware is not uniform — *many* different parameters
  - Number of caches
  - Cache sizes
  - Cache line/disk block sizes
  - Cache associativity
  - Cache replacement strategy
  - CPU/BUS/memory speed
- Programs should ideally run for many different parameters

# Algorithmic Question

- Modern hardware is not uniform — *many* different parameters
  - Number of caches
  - Cache sizes
  - Cache line/disk block sizes
  - Cache associativity
  - Cache replacement strategy
  - CPU/BUS/memory speed
- Programs should ideally run for many different parameters
  - by knowing many of the parameters at runtime
  - by knowing few essentiel parameters
  - ignoring the memory hierarchies

# Algorithmic Question

- Modern hardware is not uniform — *many* different parameters
  - Number of caches
  - Cache sizes
  - Cache line/disk block sizes
  - Cache associativity
  - Cache replacement strategy
  - CPU/BUS/memory speed

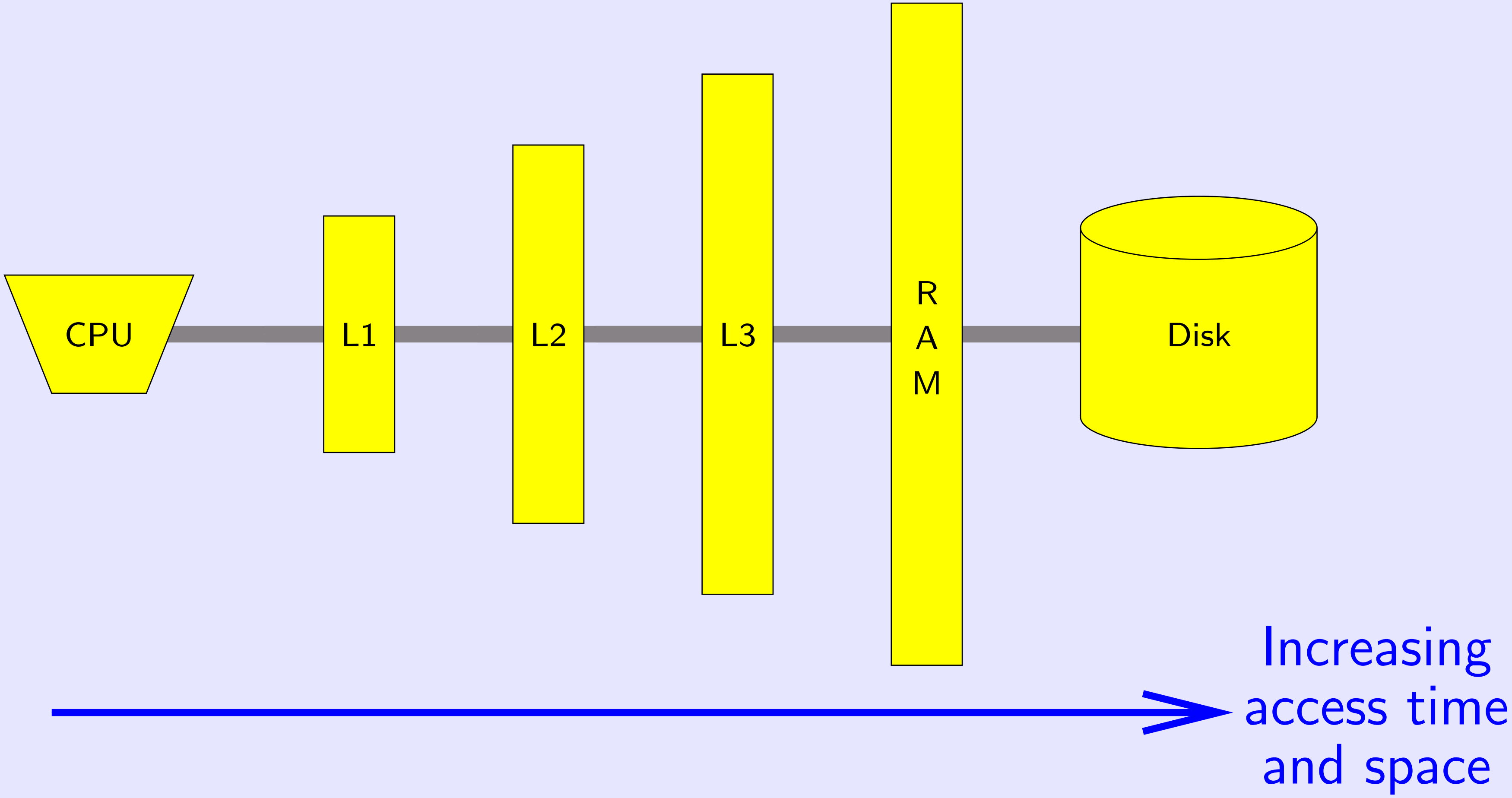
- Programs should ideally run for many different parameters
  - by knowing many of the parameters at runtime
  - by knowing few essentiel parameters
  - ignoring the memory hierarchies ← **practice**

# Algorithmic Question

- Modern hardware is not uniform — *many* different parameters
  - Number of caches
  - Cache sizes
  - Cache line/disk block sizes
  - Cache associativity
  - Cache replacement strategy
  - CPU/BUS/memory speed
- Programs should ideally run for many different parameters
  - by knowing many of the parameters at runtime
  - by knowing few essentiel parameters
  - ignoring the memory hierarchies ← **practice**
- Programs are executed on unpredictable configurations
  - Generic portable and scalable software libraries
  - Code downloaded from the internet, e.g. Java applets

# Hierarchical Memory Models

## — many parameters



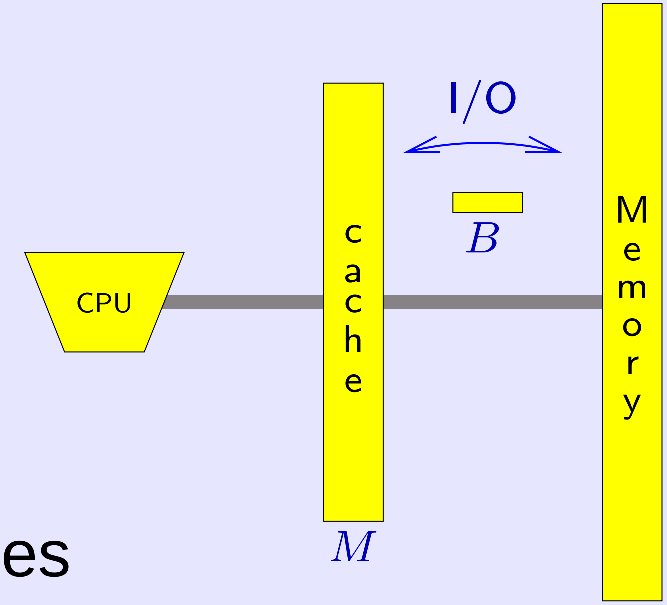
- Limited success since model to complicated

# I/O Model — two parameters

Aggarwal and Vitter 1988

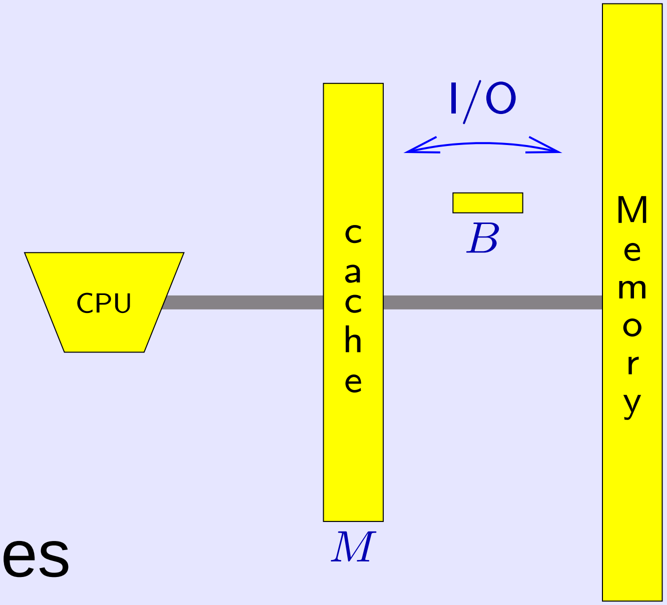
- Measure number of block transfers between two memory levels
- Bottleneck in many computations
- Very succesfull (+250 papers, many BRICS publications)
- Example: Sorting $N$ elements requires

$$O\left(\frac{N}{B}\log_{M/B}\frac{N}{M}\right) \text{ I/Os}$$



# I/O Model — two parameters

Aggarwal and Vitter 1988



- Measure number of block transfers between two memory levels
- Bottleneck in many computations
- Very succesfull (+250 papers, many BRICS publications)
- Example: Sorting $N$ elements requires

$$O\left(\frac{N}{B}\log_{M/B}\frac{N}{M}\right) \text{ I/Os}$$

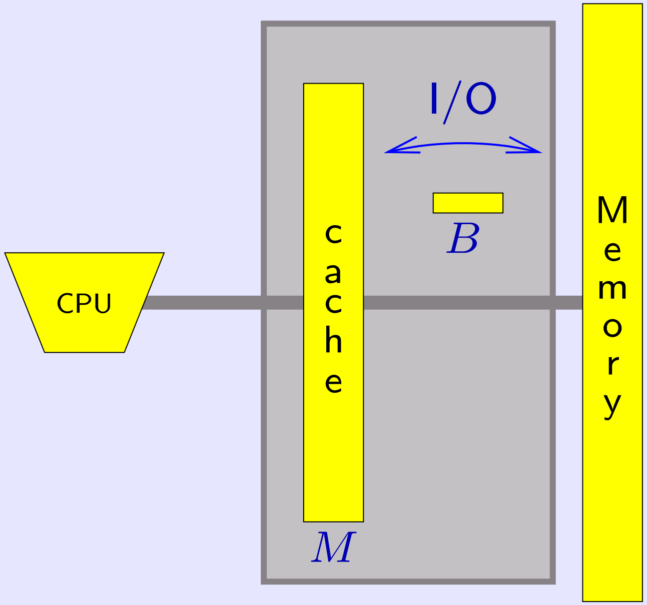
## Limitations

- Parameters $B$ and $M$ must be known
- Does not handle multiple memory levels

# Cache Oblivious Model — no parameters!?

Frigo, Leiserson, Prokop, Ramachandran 1999

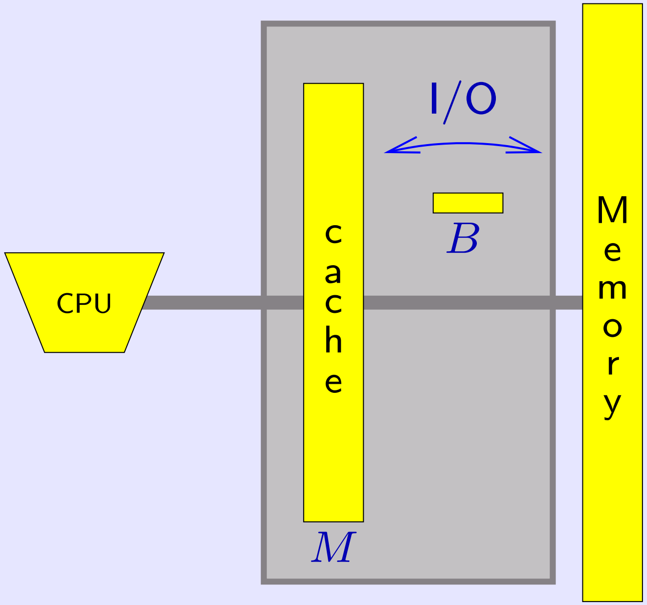
- Program with only one memory
- Analyze in the I/O model for arbitrary $B$ and $M$



# Cache Oblivious Model — no parameters!?

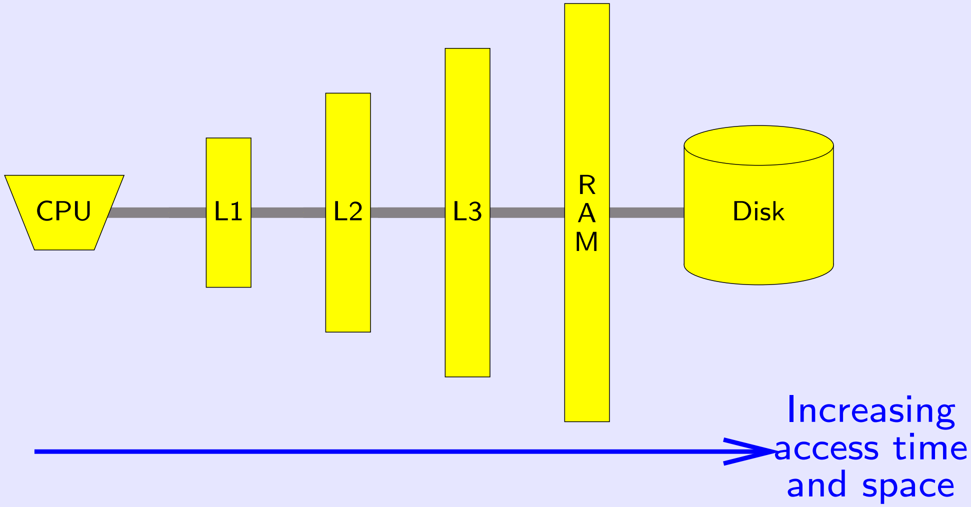
Frigo, Leiserson, Prokop, Ramachandran 1999

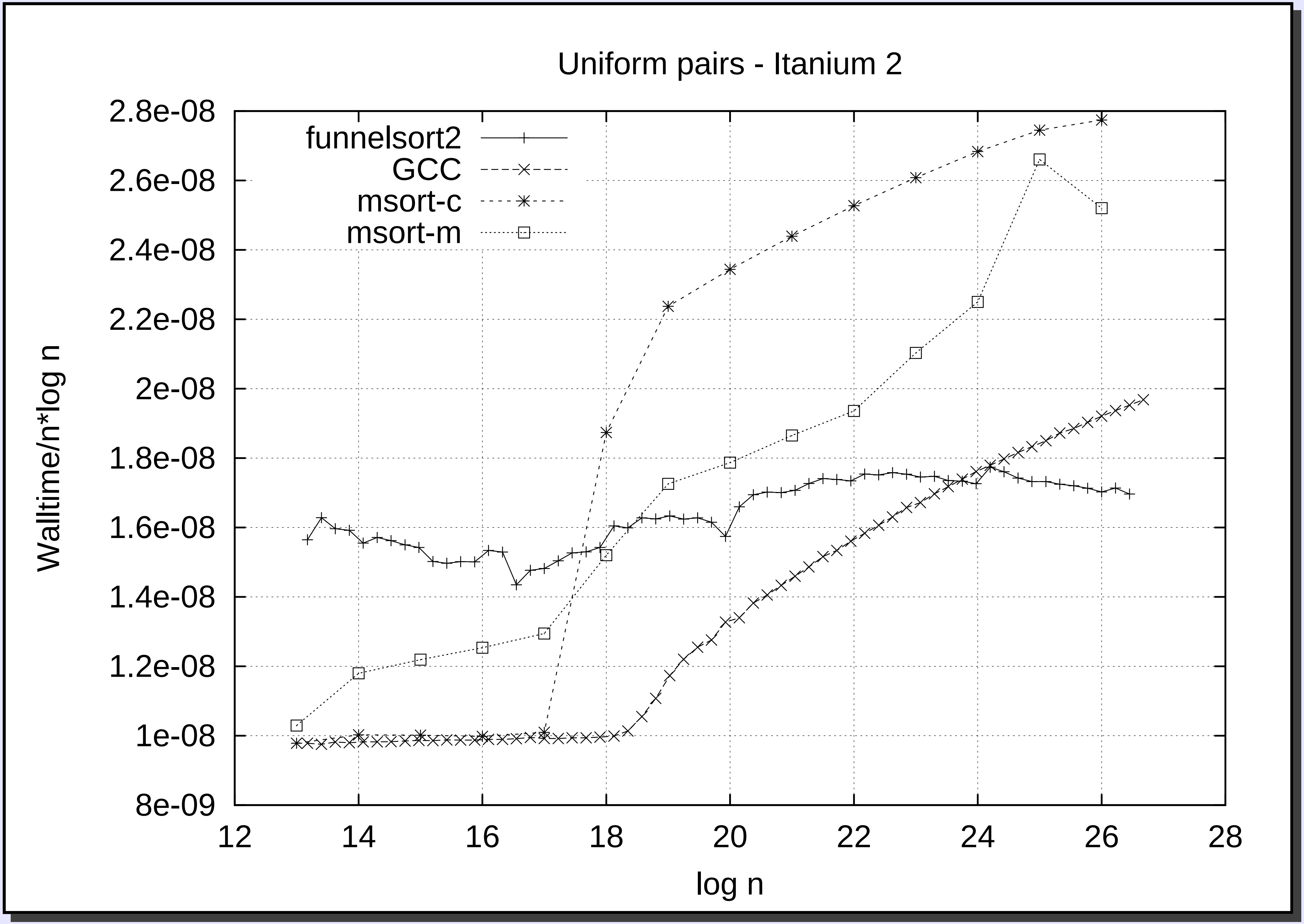
- Program with only one memory
- Analyze in the I/O model for arbitrary $B$ and $M$



## Advantages

- Optimal on arbitrary level ⇒ optimal on all levels
- Portability





*Engineering a Cache-Oblivious Sorting Algorithm*, Brodal, Fagerberg, Vinther, 2004

# Hierarchical Memory @ BRICS

- Ongoing research at BRICS since the start of BRICS
- Focus on foundational work for handling massive data sets
- Major research focus since 1998 (Brodal, Fagerberg)
- From September 2004 increased focus when Lars Arge (SNF Rømer) is joining BRICS
- BRICS publications in leading theoretical computer science conference proceedings
- Several surveys and book chapters on algorithns for massive data sets / external memory algorithms / cache-oblivious algorithms by BRICS authors
- EEF summer school on Massive data sets (2002)