BSC~Microsoft Research Centre

Vector Extensions for Decision Support DBMS Acceleration

Timothy Hayes, Oscar Palomar, Osman Unsal,
Adrian Cristal & Mateo Valero
Barcelona Supercomputing Center

Presented by **Timothy Hayes**
timothy.hayes@bsc.es

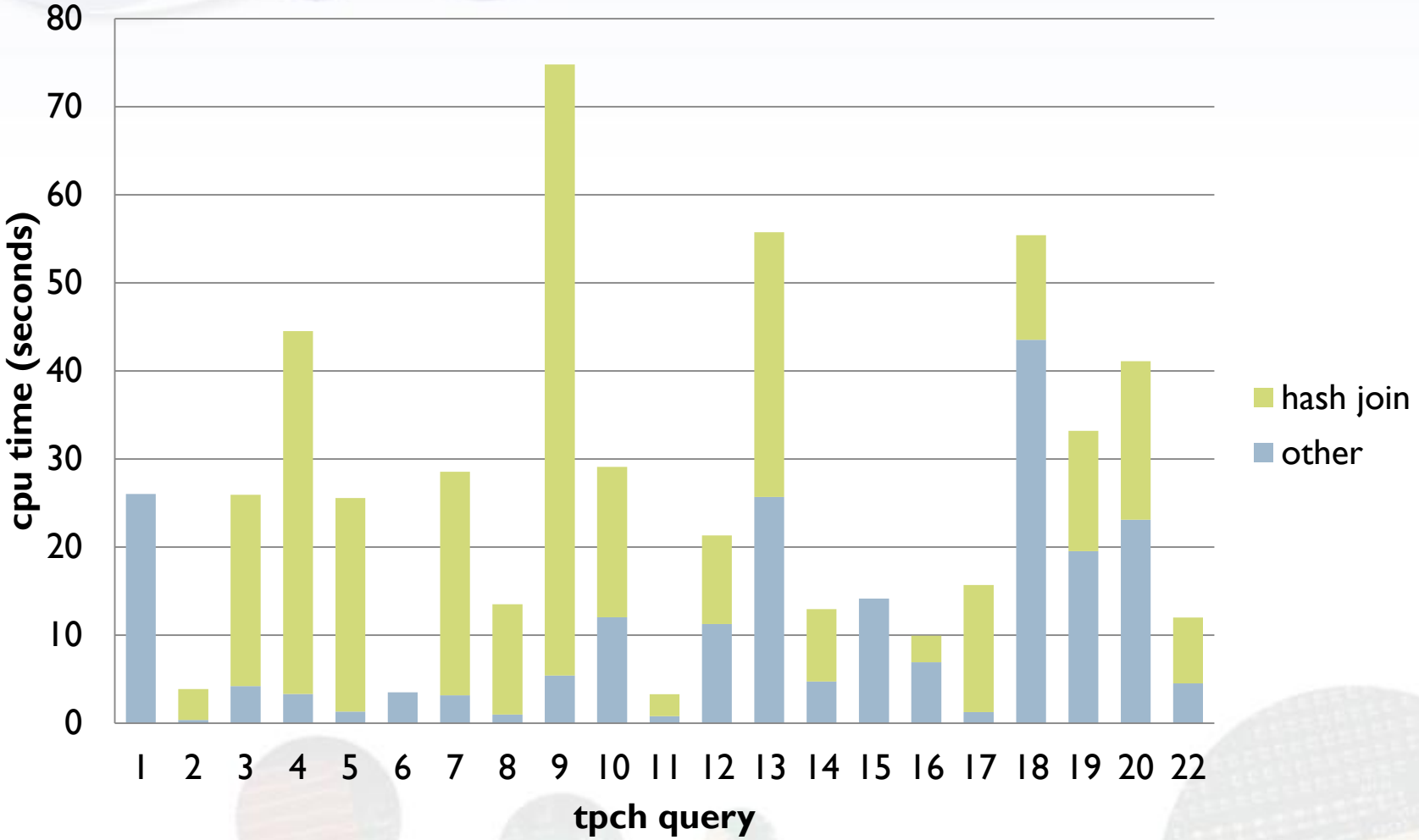
Introduction

- Databases are important
  - OnLine Analytical Processing
  - Data mining
  - E-commerce
  - Scientific analysis
- Decision Support System DBMSs
  - Extracts useful information from large structured data
  - Frequent reads – infrequent updates
  - Moved from disk-bound to memory/CPU-bound
    - Abundance of analysis
    - Recent research on DBMS implementation – *Zukowski et al (2006)*
- Opportunity for computer architecture
  - Speedup queries in a power-efficient way
  - *Data-level parallelism (DLP)* is very attractive here if available

Vectorwise

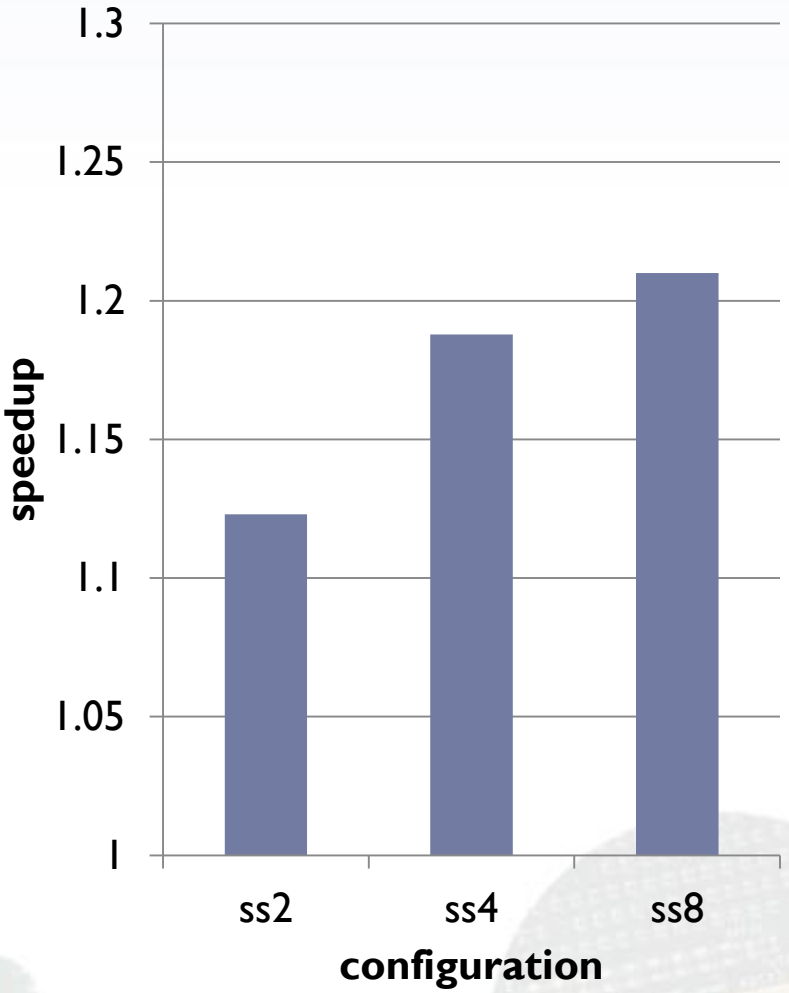
- State of the art analytical database engine
  - Based on MonetDB/X100 – *Boncz et al (2005)*
  - Redesigned database software architecture
  - Highly optimised for modern commodity superscalar CPUs
    - Finding hotspots is relevant
  - Column-oriented / block at a time (batches of values)
    - Possible opportunities for data-level parallelism (DLP)
- Profiling
  - TPC-H decision support benchmark
  - 22 queries – 100 GB database
  - Intel Nehalem microarchitecture

Profiling Vectorwise w/ TPC-H 100 GB



Hash Join Analysis

- 61 % of total time
  - Build – 33% (20%)
  - Probe – 67% (41%)
- Poor ILP scalability
  - Simulated wide configs
  - Superscalar/OoO structs
  - Maximum speedup **1.21x**
- Algorithm has DLP
  - Each probe independent
  - Why isn't it vectorised?



DLP Support in Hardware

- SIMD multimedia extensions (SSE/AVX)
  - Register lengths relatively short
  - SIMD operations are fixed in length
  - Indexed memory operations missing*
  - Experiments show speedup of less than 1%
- Vectors: traditional pipelined solution
  - Solves many problems that SIMD suffers from
    - Long vector registers with pipelined operations
    - Programmable vector length
    - Mask registers for conditionals
    - Gather/scatter
  - Traditionally applied to scientific/multimedia domains
    - Opportunity to explore business-domain applications

Paper Contributions

- Show that a vectors can be reapplied to DSS workloads
- Extend modern out-of-order x86-64 microprocessor
  - Provide suitable vector ISA extensions
  - Optimise implementation for DSS workload
- Experimental evaluation
  1. Demonstrate that vectors are beneficial
  2. Design space exploration
  3. Memory bandwidth analysis
  4. Prefetching support

Vector Extensions to x86-64

- Vector Instruction Set
  - Traditional instructions
    - Vectorises hash join
    - But not overly specific
  - Integer over floating point
  - Classes
    - Arithmetic / Logical
    - Compress
    - Optional Mask
      - Mask Arithmetic
    - Programmable Vector Length
    - Mem. Unit Stride / Indexed
- Architecture
  - **8** vector registers
    - Size discussed later
  - **4** mask registers
  - **1** vector length register
- Microarchitecture
  - Adds **3** new vector clusters
    - **2** arithmetic - **1** memory
  - Tightly integrated with core
    - *Not* a coprocessor
    - Reuse existing structures
    - Cache integration difficult
    - OoO difficult

Cache Hierarchy Integration

- Want to take advantage of the cache hierarchy
  - Vectorwise is blocked & cache-conscious
  - Sometimes datasets are cache-resident
- Vector integration should...
  - Not compromise the existing access time of the L1D cache
  - Provide enough bandwidth to vector unit
  - Exploit regular access patterns, i.e. unit stride
- Bypass L1D and go directly to L2
  - *Quintana et al. (1999)*
  - Pull many elements in a single request
  - Amortise extra latency incurred w/ long pipelined ops

Out of Order Execution

- *Espasa et al. (1997)* vectors with out of order execution
  - Performance benefits ✔
  - Hides memory latency even more ✔
  - Only supports unit-stride memory access ✘
- Very difficult for indexed accesses
  - Need to check for memory aliases
  - Gather/Scatter too complex for load/store queue (LSQ)
- Our proposal
  - Explicitly program fences between memory dependencies
    - Seldomly needed
  - Relax the memory model
  - Bypass the LSQ completely
  - Very simple hardware to track outstanding memory ops

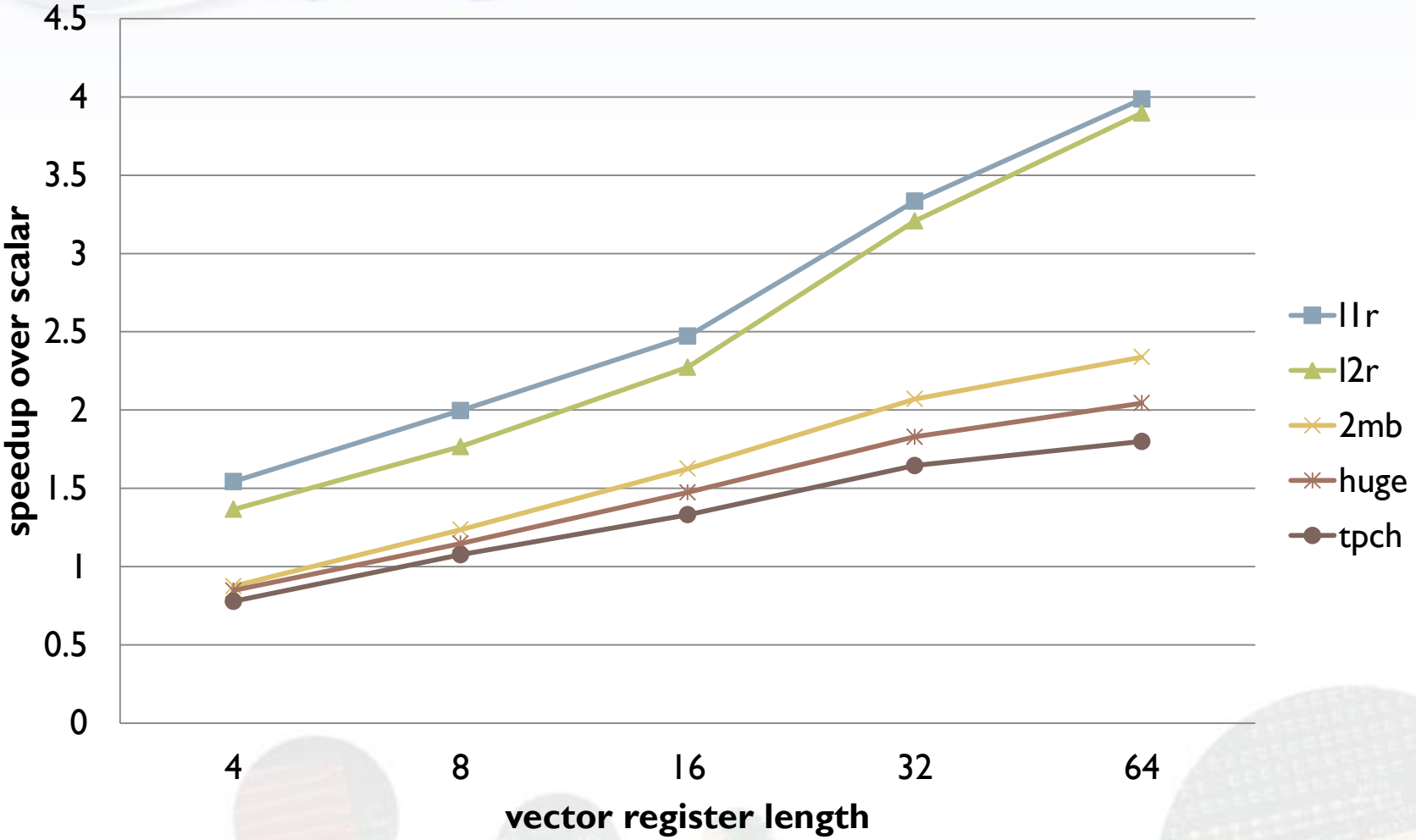
Experimental Setup

- Scalar Baseline
  - Intel Nehalem 2.67 GHz
  - Single core
  - Inclusive Cache
    - L1i – 32 KB – 1 cycle
    - L1d – 32 KB – 4 cycles
    - L2 – 256 KB – 10 cycles
- Memory System
  - DDR3-1333
  - 10.667 GB/s bandwidth
- Simulators
  - PTLsim
  - DRAMSim2
- Application
  - Hand-vectorised
- Datasets
  1. L1 resident (l1r)
  2. L2 resident (l2r)
  3. 2 MB
  4. HUGE
  5. TPCH

Vector Benefits

Are vectors suitable for DSS acceleration?

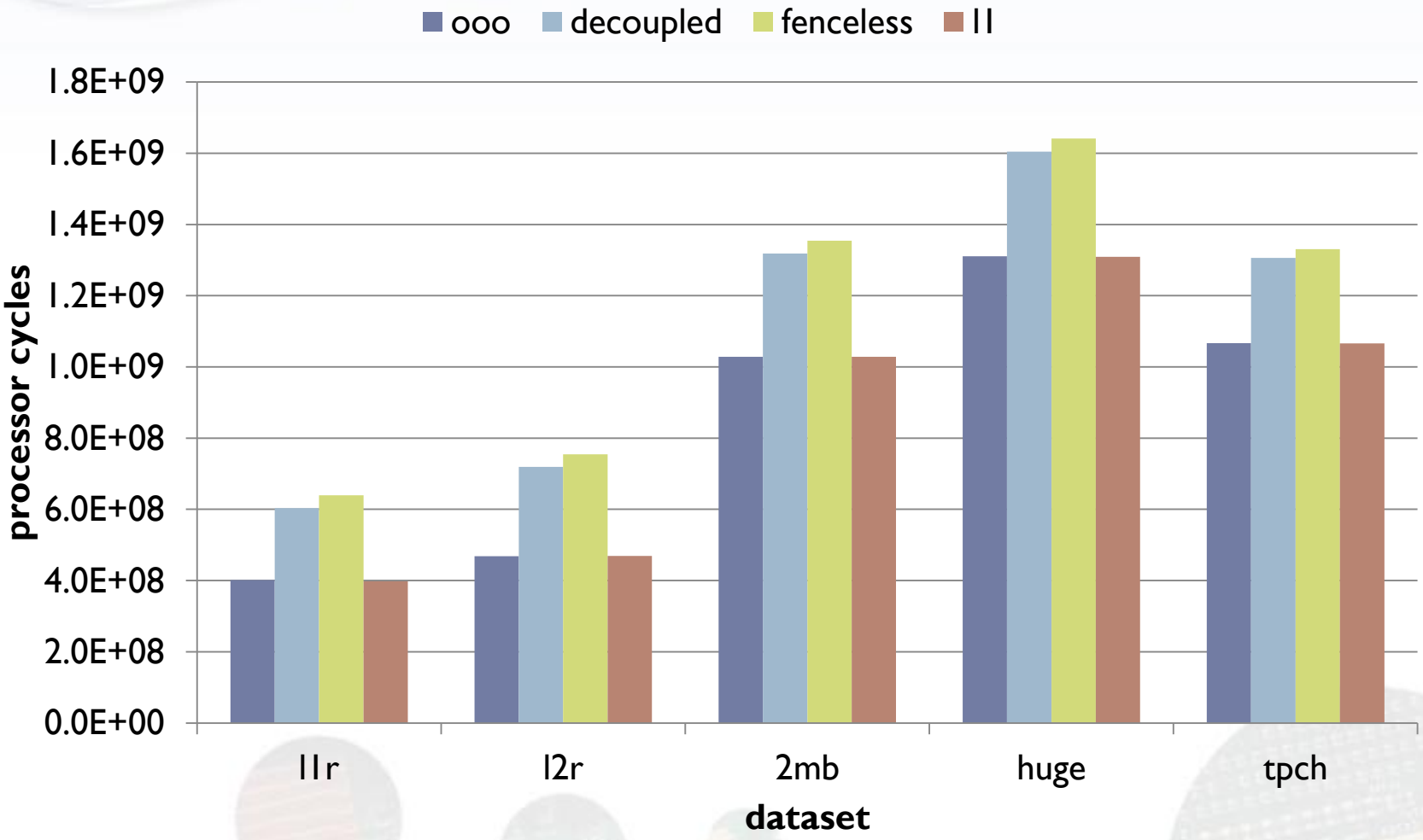
Scalability of Vector Length



Design Exploration

Are the design decisions justified?

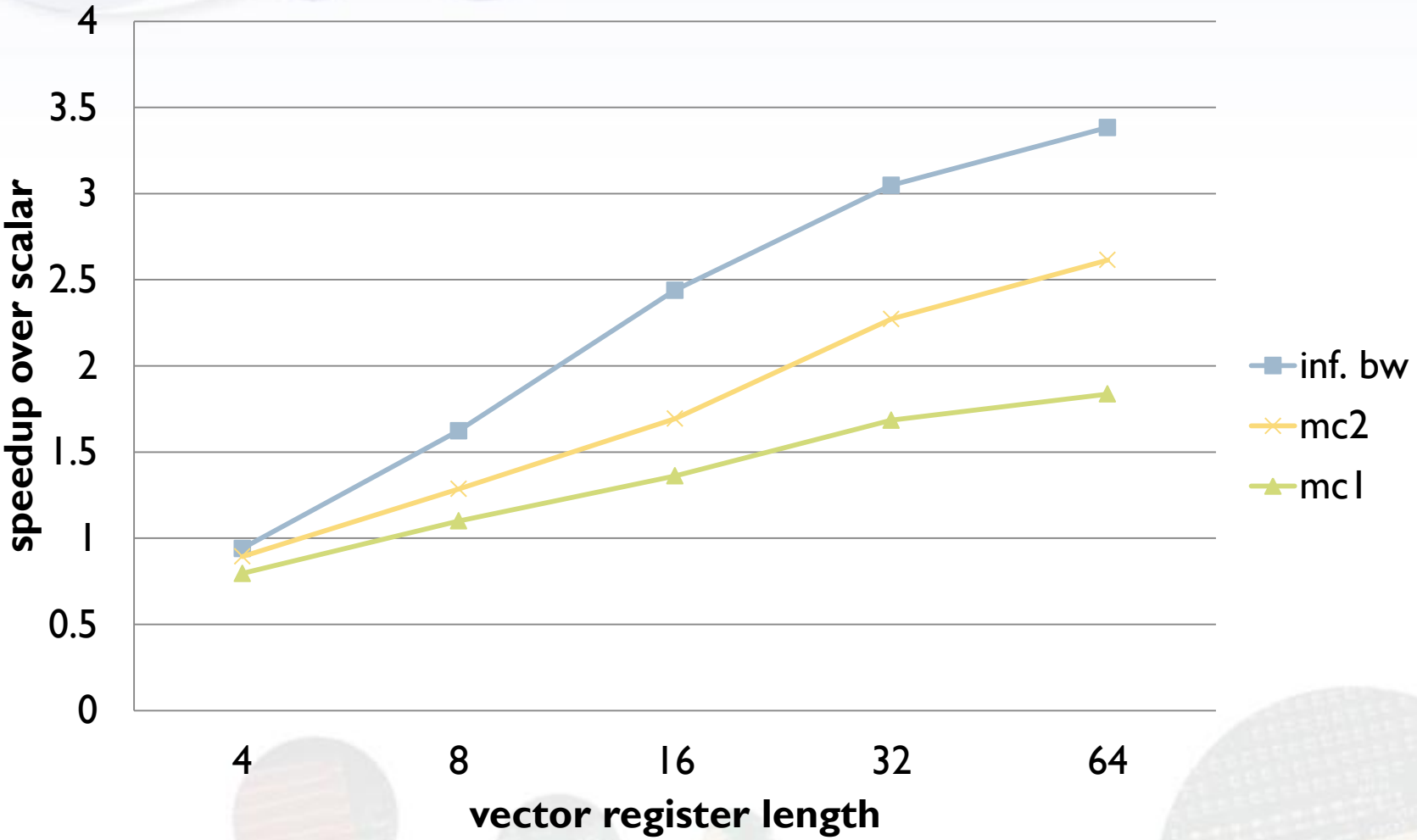
Design Exploration – MVL64



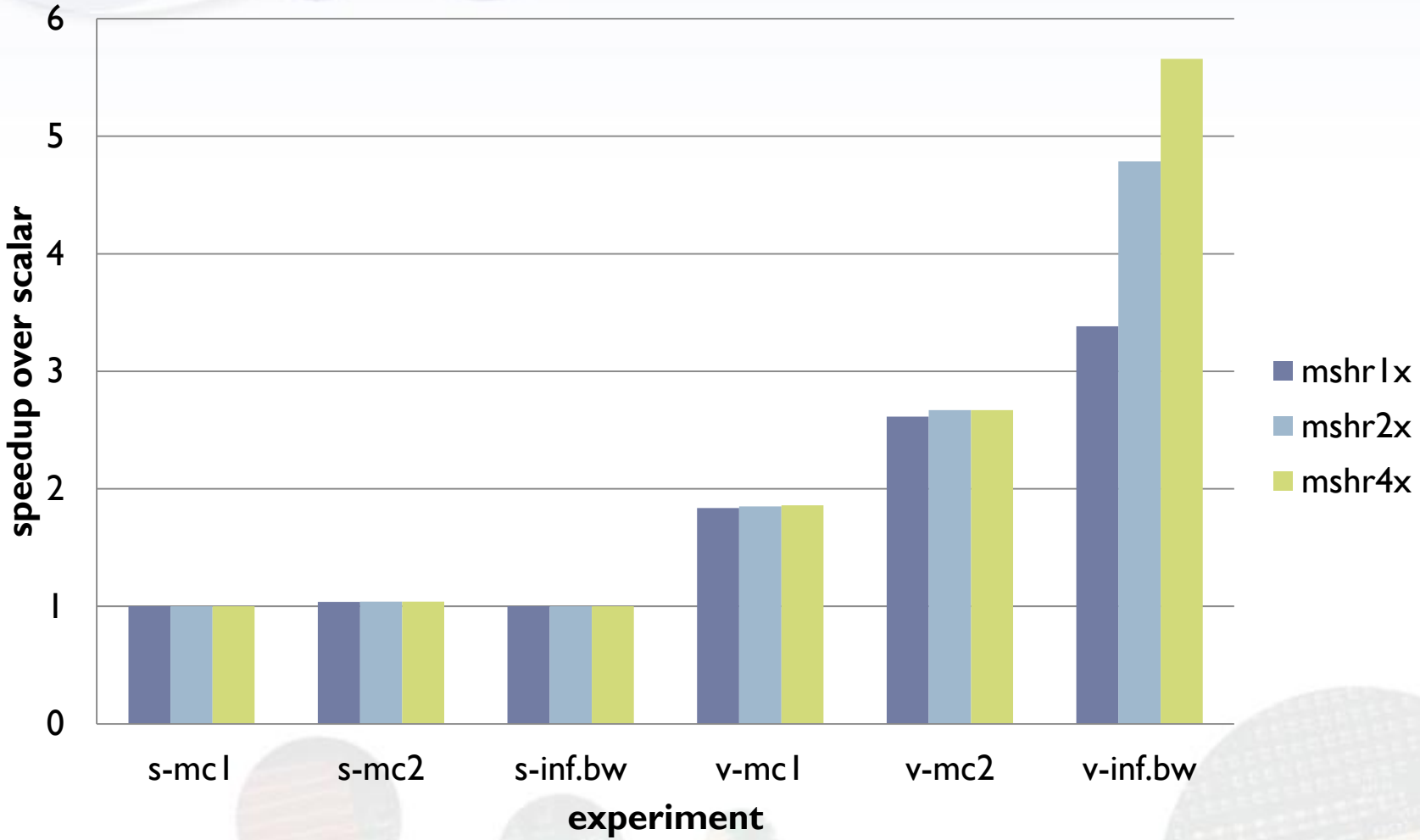
Memory Bandwidth

Can vectors utilise the available bandwidth?

Memory Bandwidth Utilisation



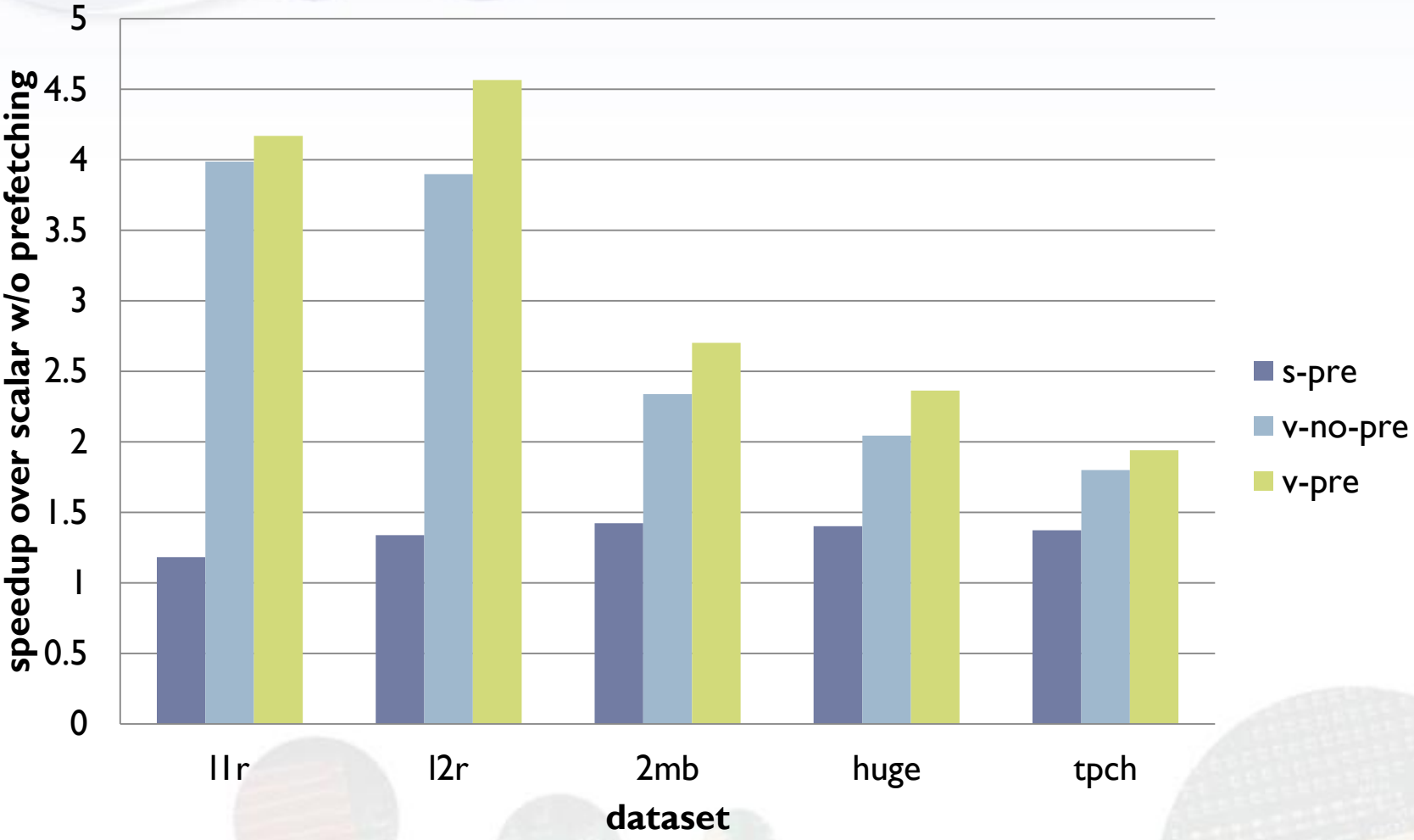
Memory Bandwidth / MSHRs – MVL64



Software Prefetching Support

Increasing the utilisation of available memory bandwidth

Prefetching Improvements – MVL64



Conclusions

- Superscalar/OoO
  - Does not offer good scalability for a DSS workload
  - Does not saturate available memory bandwidth
- Vectors ideal for a DSS workload
  - Speedup between **1.94x** – **4.56x** for **41%** of benchmark
  - Fully saturates available memory bandwidth
  - Long vector operations
    - Potential to scale further
    - All with *pipelining* and not parallel lanes
- Design Space Measurements
  - Cache integration
    - Bypassing L1 cache does not incur a penalty
  - Out of order integration
    - Indexed memory support is challenging
    - **1.4x** improvement
    - Future work will discover its cost in area/energy

BSC~Microsoft Research Centre

Vector Extensions for Decision Support DBMS Acceleration

Timothy Hayes, Oscar Palomar, Osman Unsal,
Adrian Cristal & Mateo Valero
Barcelona Supercomputing Center

Presented by **Timothy Hayes**
timothy.hayes@bsc.es