

并行计算

Parallel Computing

主讲人 孙广中

Spring, 2018

思考题1

- 问题1:

谈谈你所知道的高性能计算与云计算的区别?

- 问题2:

并行程序的描述应如何? 与串行程序有什么不同?

- 问题3:

如何并行地尽快求解 n 个元素的最大值或排序?

- 第一篇 并行计算硬件平台：并行计算机
 - 第一章 并行计算与并行计算机结构模型
 - 第二章 并行计算机系统互连与基本通信操作
 - 第三章 典型并行计算机系统介绍
 - 第四章 并行计算性能评测

第一章 并行计算及并行机结构模型

- 1.1 计算与计算机科学
- 1.2* 单处理机与指令级并行
- 1.3* 多核处理器与线程级并行
- 1.4 并行计算机体系结构
 - 1.4.1 并行计算机结构模型
 - 1.4.2 并行计算机访存模型
 - 1.4.3* 并行计算机存储模型
- 1.5 并行计算概述

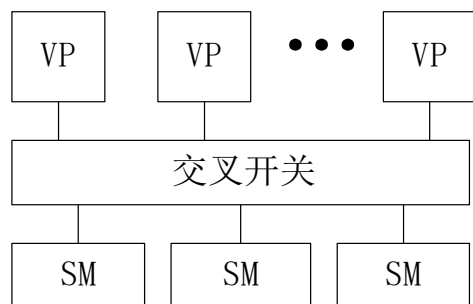
并行计算、计算科学、计算需求

- 并行计算：并行机上所作的计算，又称高性能计算或超级计算。
- 计算科学：计算物理、计算化学、计算生物等。
- 计算是科学发现的三大支柱之一。
- 科学与工程问题的需求：气象预报、油藏模拟、核武器数值模拟、航天器设计、基因测序等。
 - 美国ASCI计划(1996)：核武器数值模拟。
- 需求类型：计算密集、数据密集、网络密集。

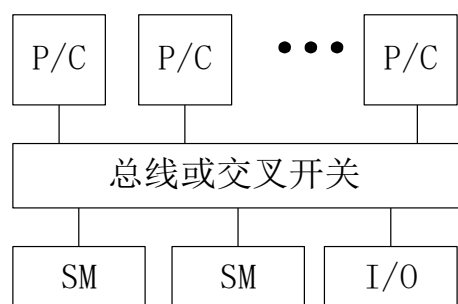
第一章 并行计算及并行机结构模型

- 1.1 计算与计算机科学
- 1.2* 单处理机与指令级并行
- 1.3* 多核处理器与线程级并行
- 1.4 并行计算机体系结构
 - 1.4.1 并行计算机结构模型
 - 1.4.2 并行计算机访存模型
 - 1.4.3* 并行计算机存储模型
- 1.5 并行计算概述

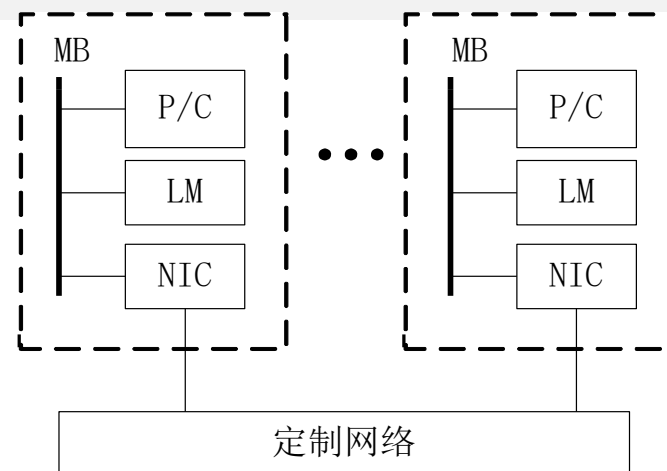
并行计算机结构模型 (1)



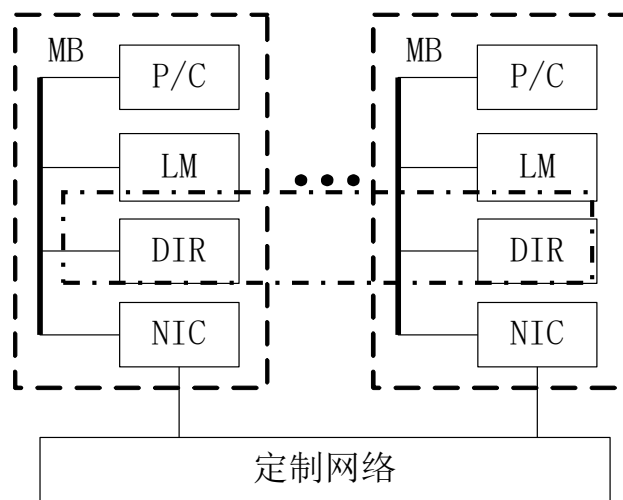
(a) PVP



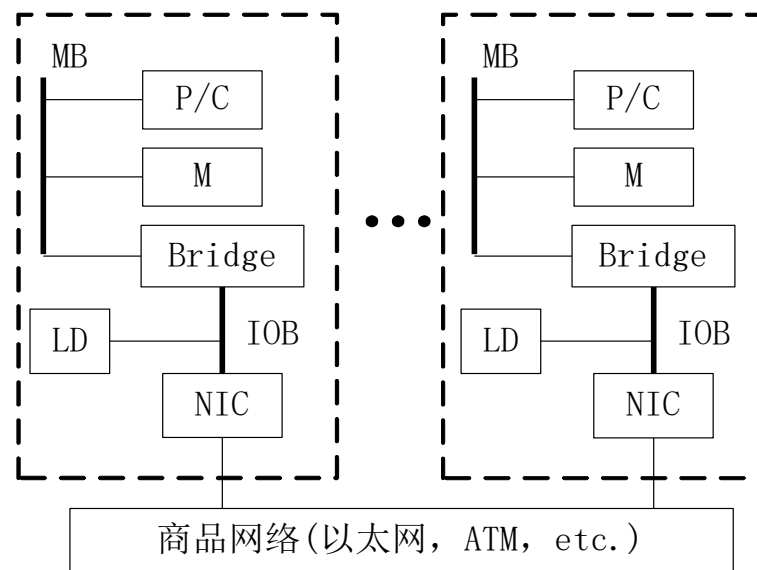
(b) SMP



(c) MPP

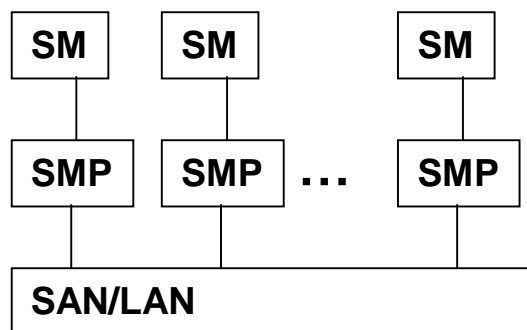


(d) DSM

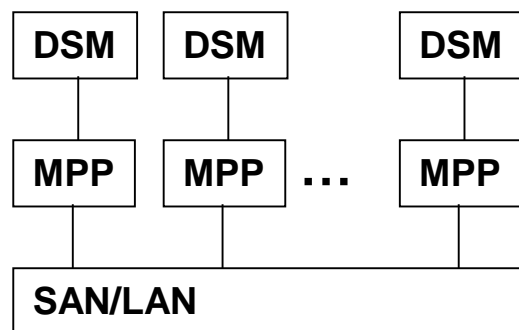


(e) COW

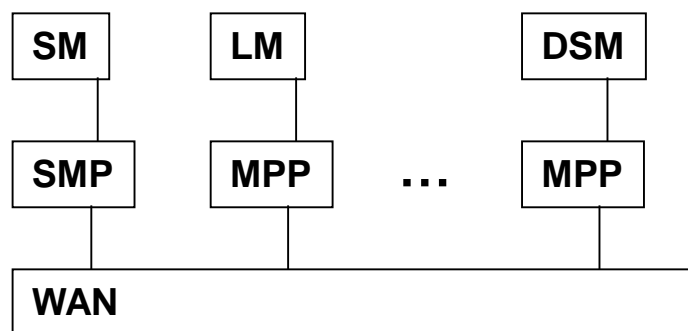
并行计算机结构模型 (2)



(f) SMP-Cluster



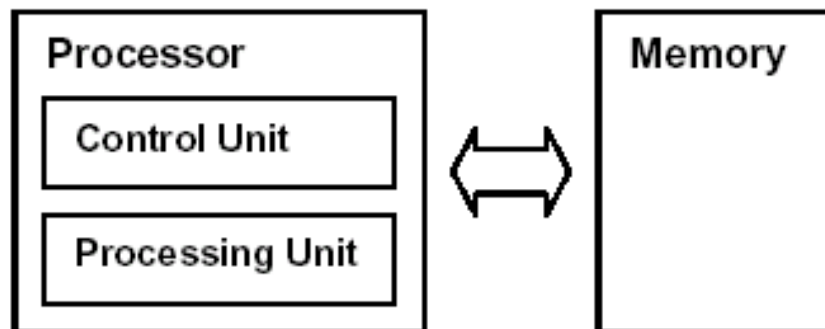
(g) DSM-Cluster



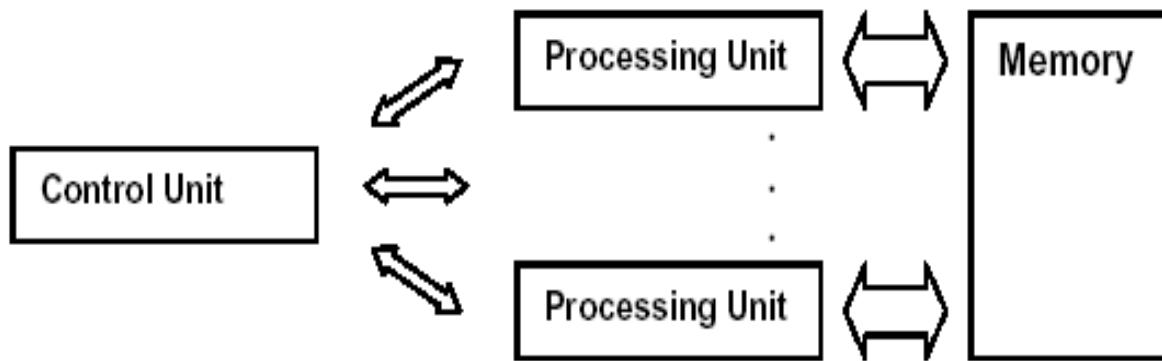
(h) Grid (Cluster of Clusters)

并行计算机结构模型（3）

SISD computer -Von Neumann's model

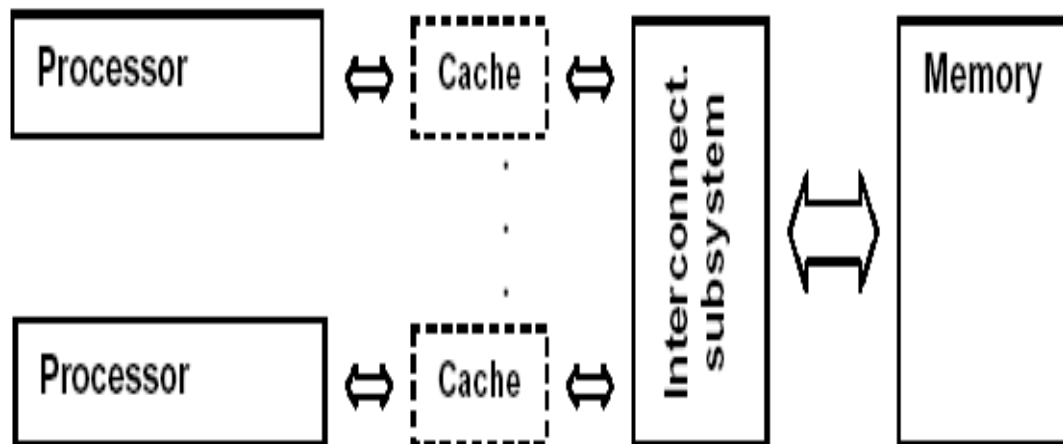


SIMD computer

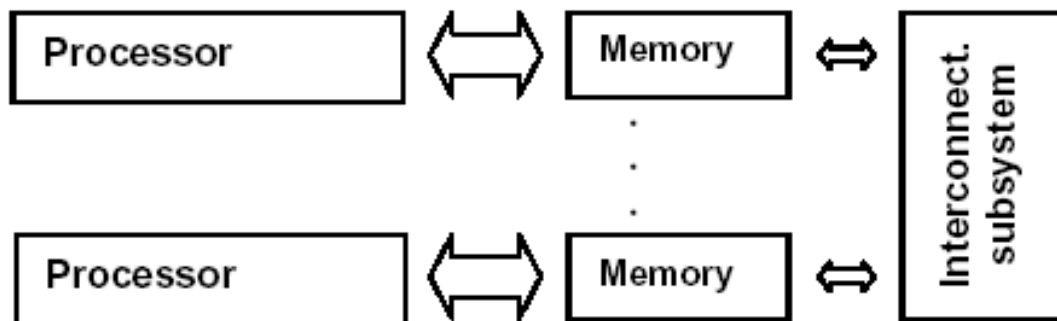


并行计算机结构模型（4）

Symmetric multiprocessor – MIMD-SM

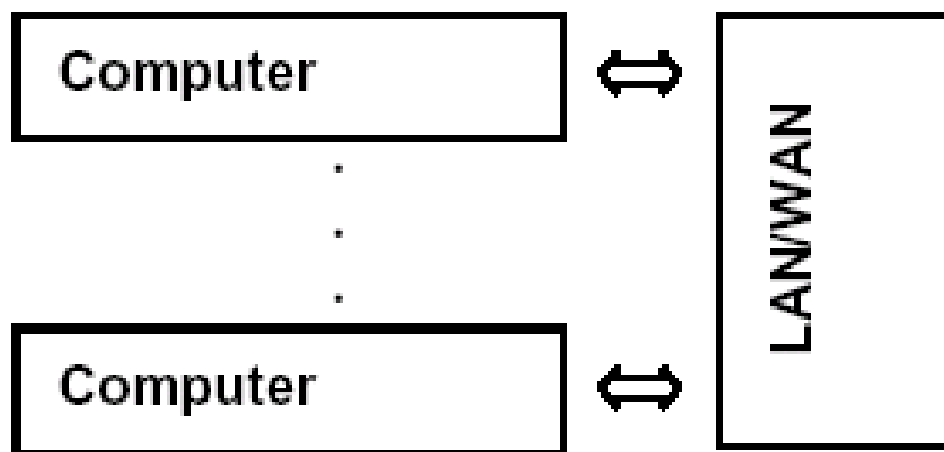


Massively parallel processor – MIMD-DM



并行计算机结构模型（5）

Cluster of workstations – MIMD-DM

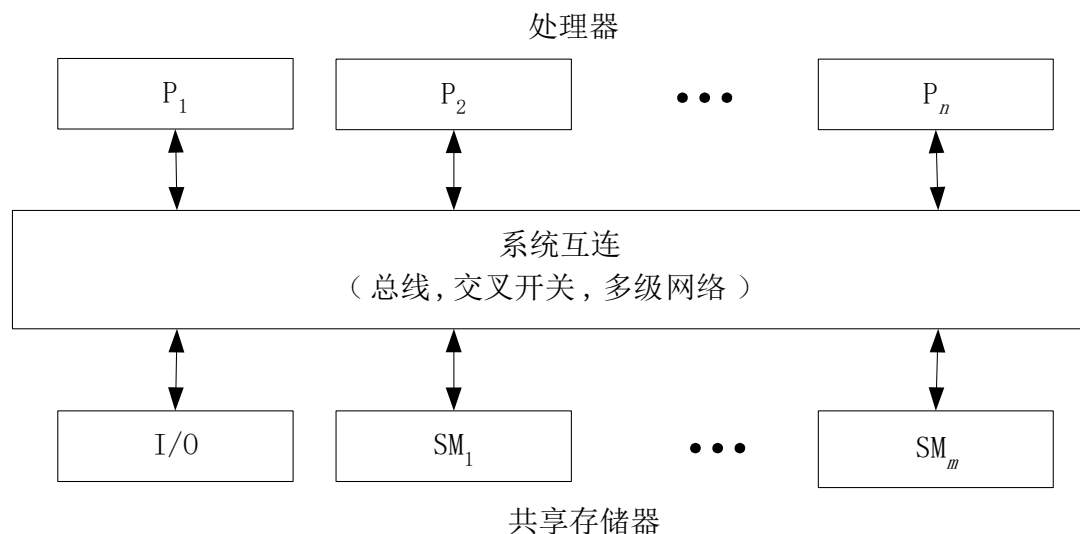


并行计算机结构模型（6）——五种结构小结

属性	PVP	SMP	MPP	DSM	COW
结构类型	MIMD	MIMD	MIMD	MIMD	MIMD
处理器类型	专用定制	商用	商用	商用	商用
互连网络	定制交叉开关	总线、交叉开关	定制网络	定制网络	商用网络（以太ATM）
通信机制	共享变量	共享变量	消息传递	共享变量	消息传递
地址空间	单地址空间	单地址空间	多地址空间	单地址空间	多地址空间
系统存储器	集中共享	集中共享	分布非共享	分布共享	分布非共享
访存模型	UMA	UMA	NORMA	NUMA	NORMA
代表机器	Cray C-90, Cray T-90, 银河1号	IBM R50, SGI Power Challenge, 曙光1号	Intel Paragon, IBMSPP2, 曙光1000/2000	Stanford DASH, Cray T 3D	Berkeley NOW, Alpha Farm

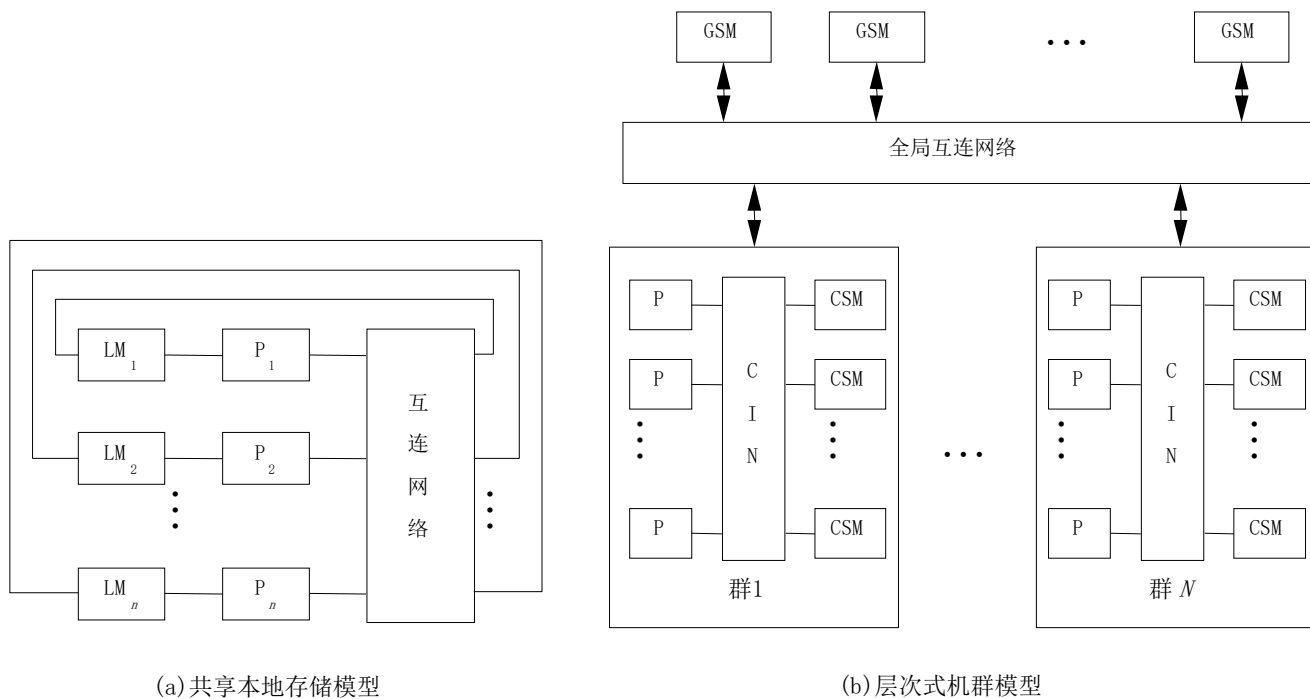
并行计算机访存模型（1）——UMA

- UMA（Uniform Memory Access）模型是均匀存储访问模型的简称。其特点是：
 - 物理存储器被所有处理器均匀共享；
 - 所有处理器访问任何存储字取相同的时间；
 - 每台处理器可带私有高速缓存；
 - 外围设备也可以一定形式共享。



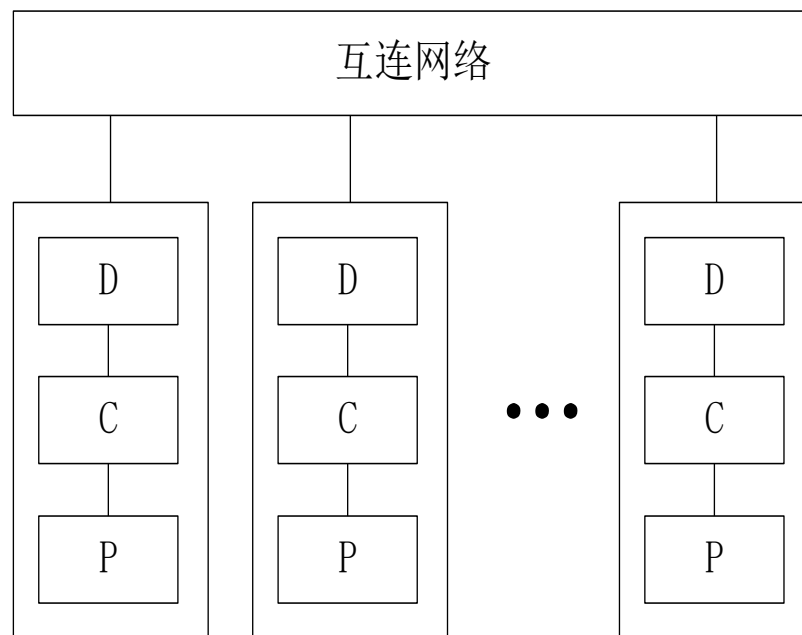
- NUMA(Nonuniform Memory Access)模型是非均匀存储访问模型的简称。特点是：
 - 被共享的存储器在物理上是分布在所有的处理器中的，其所有本地存储器的集合就组成了全局地址空间；
 - 处理器访问存储器的时间是不一样的；访问本地存储器LM或群内共享存储器CSM较快，而访问外地的存储器或全局共享存储器GSM较慢(此即非均匀存储访问名称的由来)；
 - 每台处理器照例可带私有高速缓存，外设也可以某种形式共享。

并行计算机访存模型（2）——NUMA



- COMA(Cache-Only Memory Access)模型是**全高速缓存存储访问**的简称。其特点是：
 - 各处理器节点中没有存储层次结构，全部高速缓存组成了全局地址空间；
 - 利用分布的高速缓存目录D进行远程高速缓存的访问；
 - COMA中的高速缓存容量一般都大于2 级高速缓存容量；
 - 使用COMA时，数据开始时可任意分配，因为在运行时它最终会被迁移到要用到它们的地方。

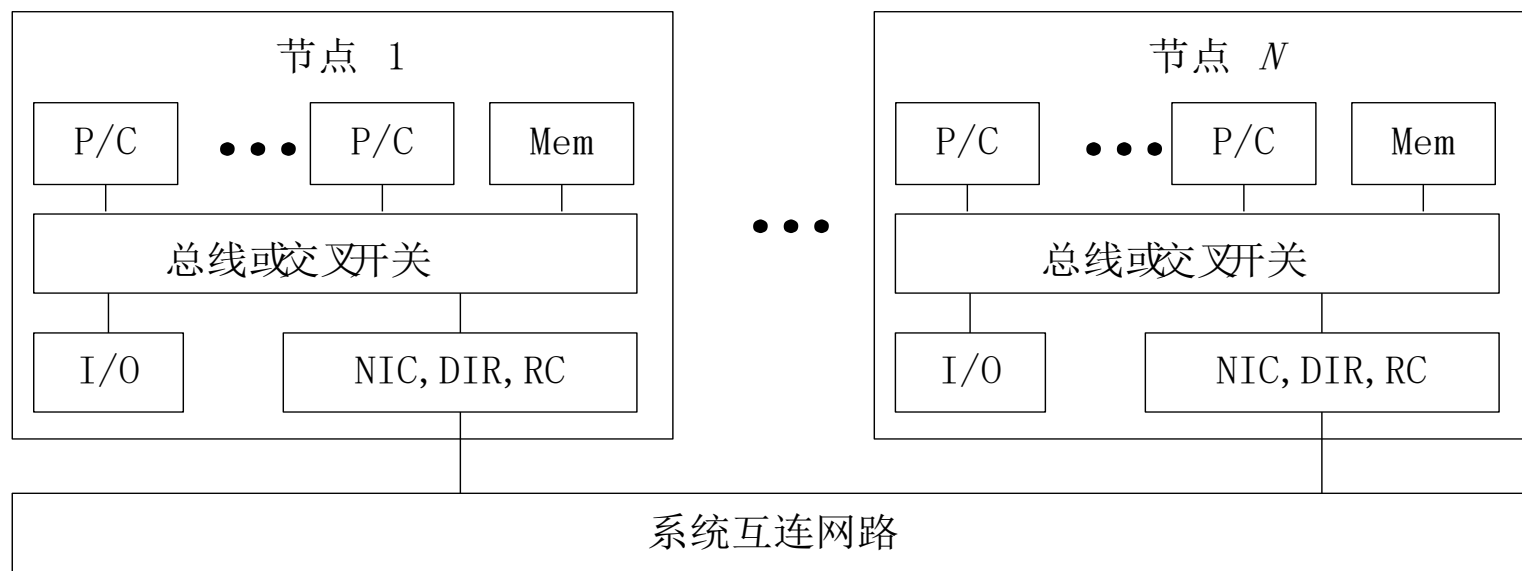
并行计算机访存模型（3）——COMA



并行计算机访存模型（4）——CC-NUMA

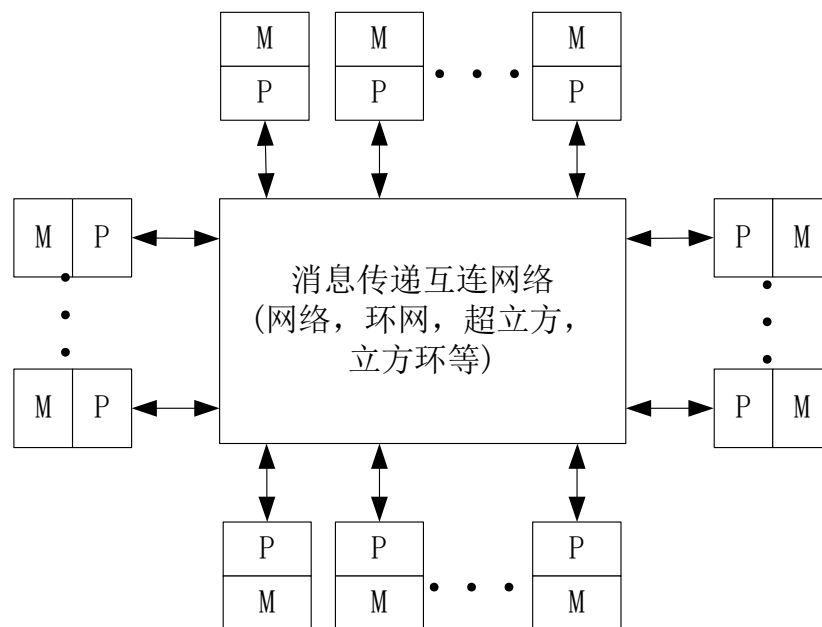
- CC-NUMA（Coherent-Cache Nonuniform Memory Access）模型是**高速缓存一致性非均匀存储访问模型**的简称。其特点是：
 - 大多数使用基于目录的高速缓存一致性协议；
 - 保留SMP结构易于编程的优点，也改善常规SMP的可扩放性；
 - CC-NUMA实际上是一个分布共享存储的DSM多处理机系统；
 - 它最显著的优点是程序员无需明确地在节点上分配数据，系统的硬件和软件开始时自动在各节点分配数据，在运行期间，高速缓存一致性硬件会自动地将数据迁移至要用到它的地方。

并行计算机访存模型（4）——CC-NUMA

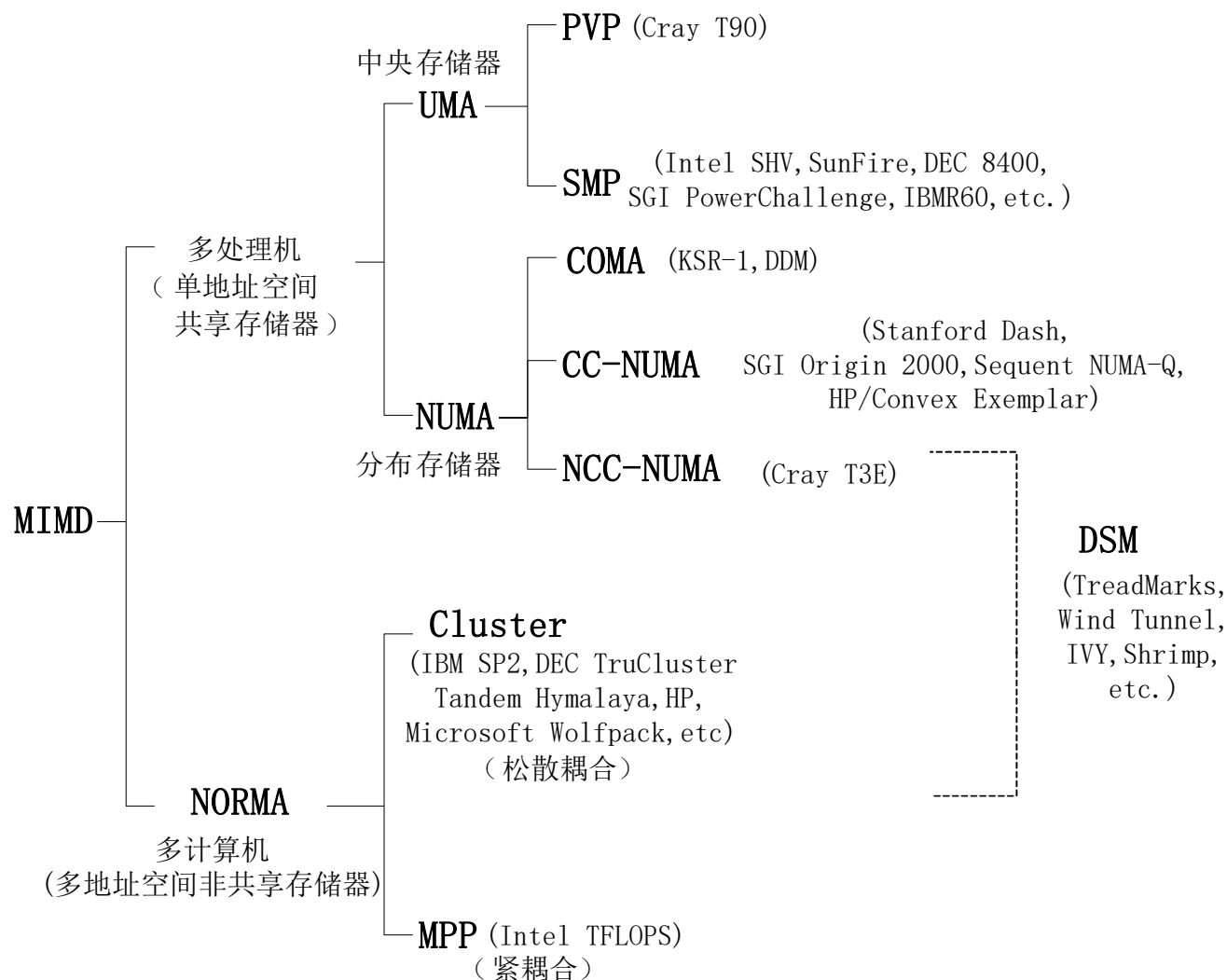


并行计算机访存模型（5）——NORMA

- NORMA (No-Remote Memory Access) 模型是**非远程存储访问模型**的简称。NORMA的特点是：
 - 所有存储器是私有的，仅能由其处理器访问；
 - 绝大数NORMA都不支持远程存储器的访问；



并行机系统的不同存储结构



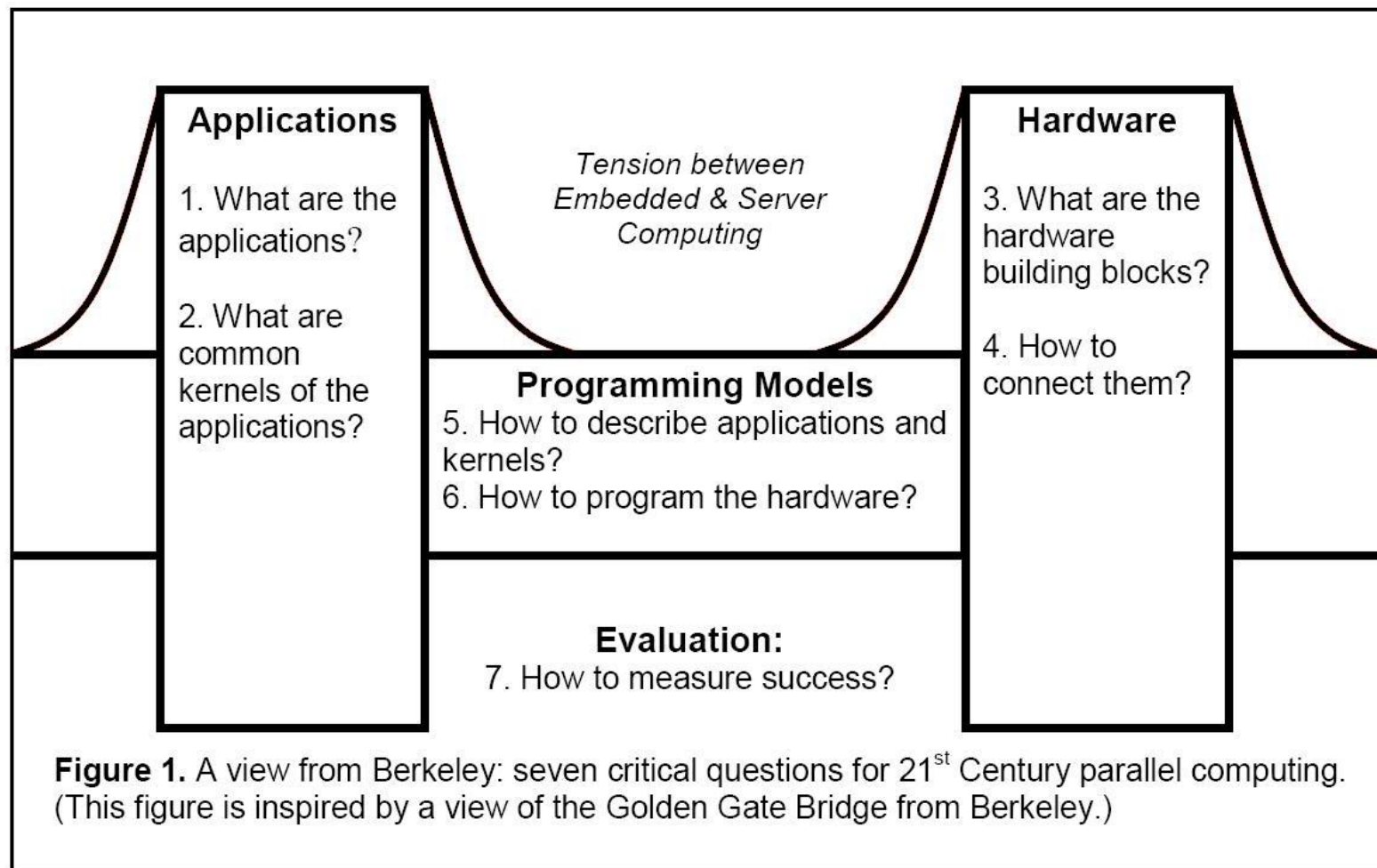
第一章 并行计算及并行机结构模型

- 1.1 计算与计算机科学
- 1.2* 单处理机与指令级并行
- 1.3* 多核处理器与线程级并行
- 1.4 并行计算机体系结构
 - 1.4.1 并行计算机结构模型
 - 1.4.2 并行计算机访存模型
 - 1.4.3* 并行计算机存储模型
- 1.5 并行计算概述

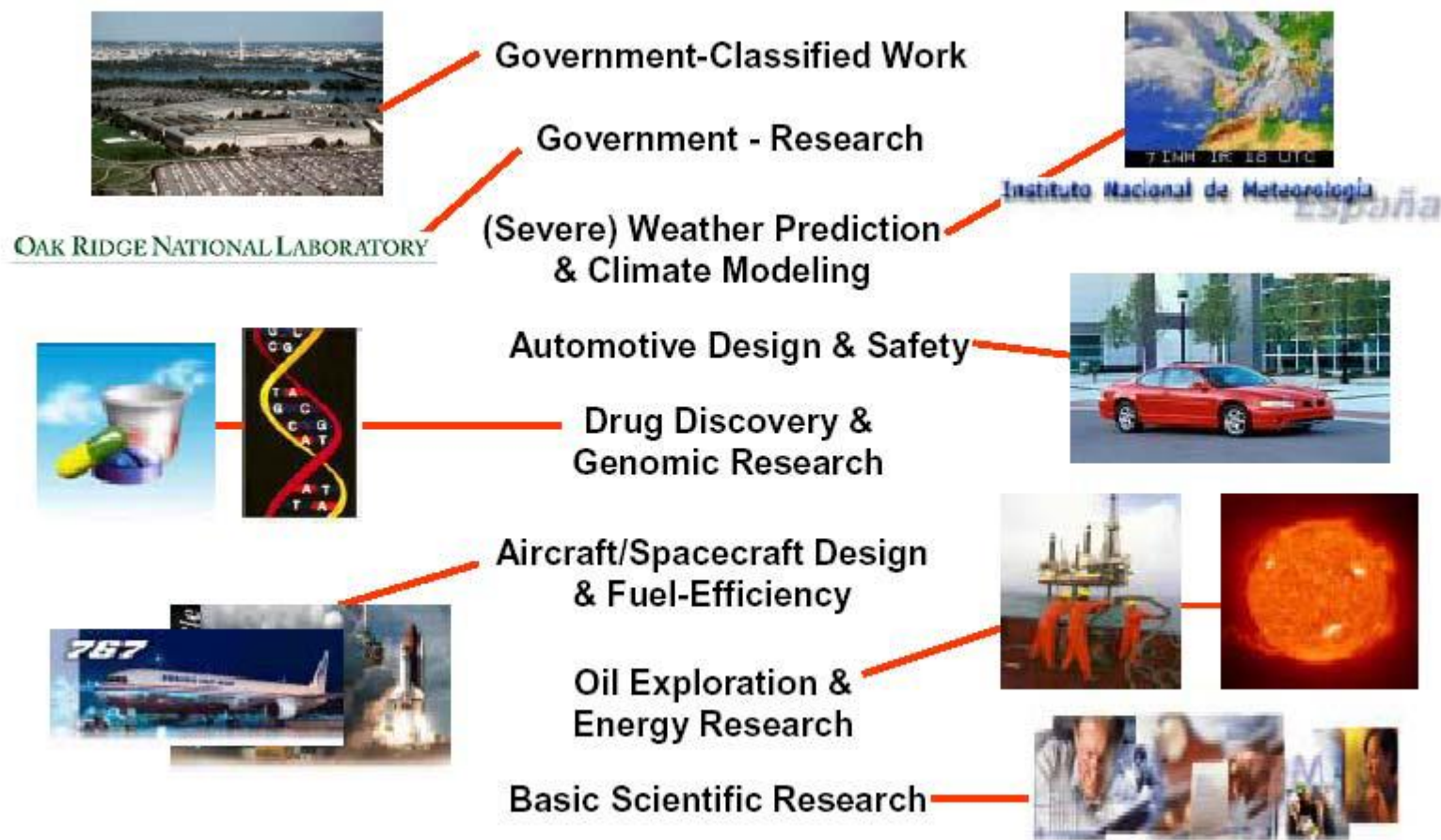
The Landscape of Parallel Computing Research

- A view from Berkeley
 - 2006年的白皮书
 - 计算数学、计算物理、计算机软件、计算机硬件、计算机算法、电子学等多个学科20余名教授，半年时间研讨
 - http://view.eecs.berkeley.edu/wiki/Main_Page

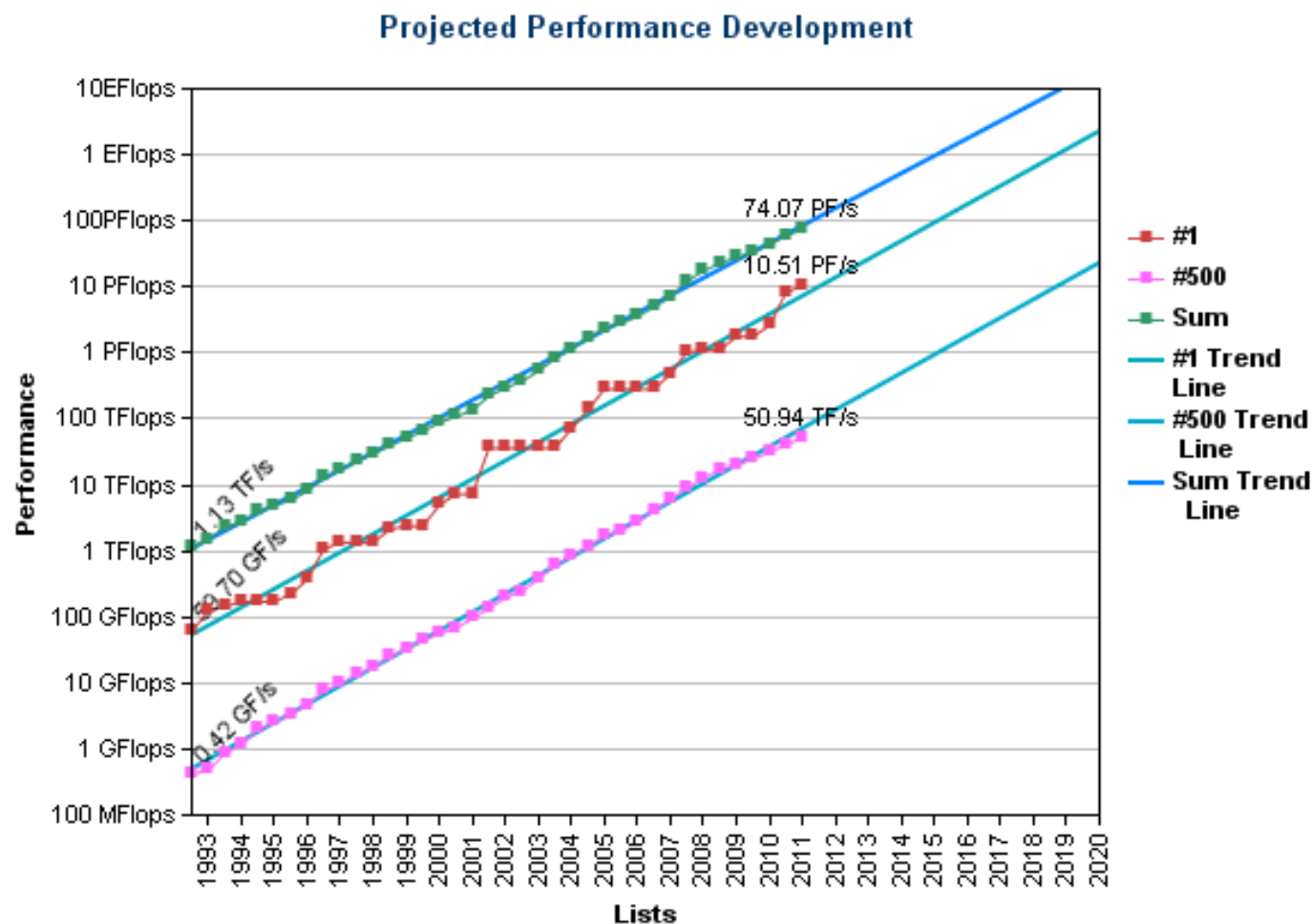
A view from Berkeley



高性能计算应用需求



超级计算机性能的发展趋势



作业1（研究生必做，本科生选做）

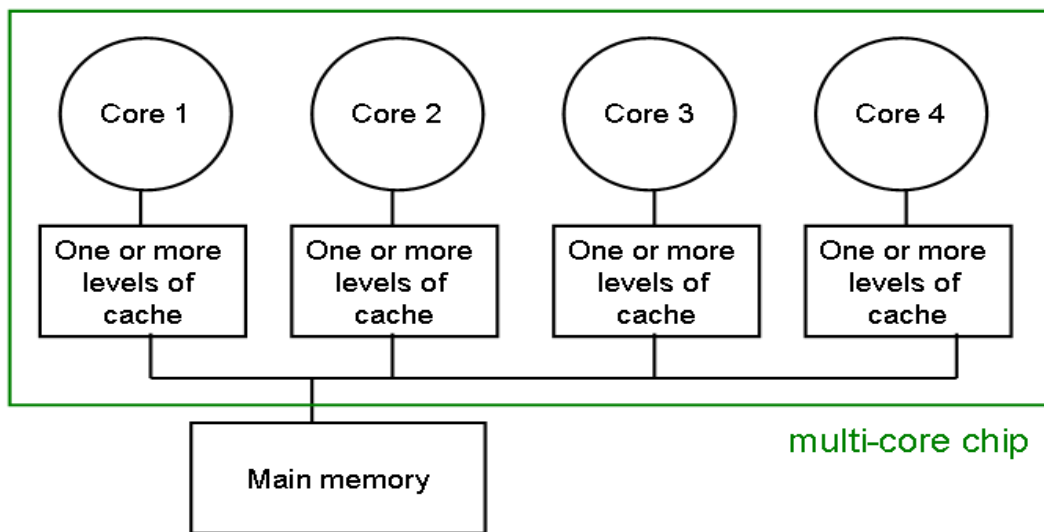
1. 课本第35页1.11题。并行计算应用调研。
2. 根据调研的应用需求，预算150万人民币购置计算资源。请确定购买的机器配置。

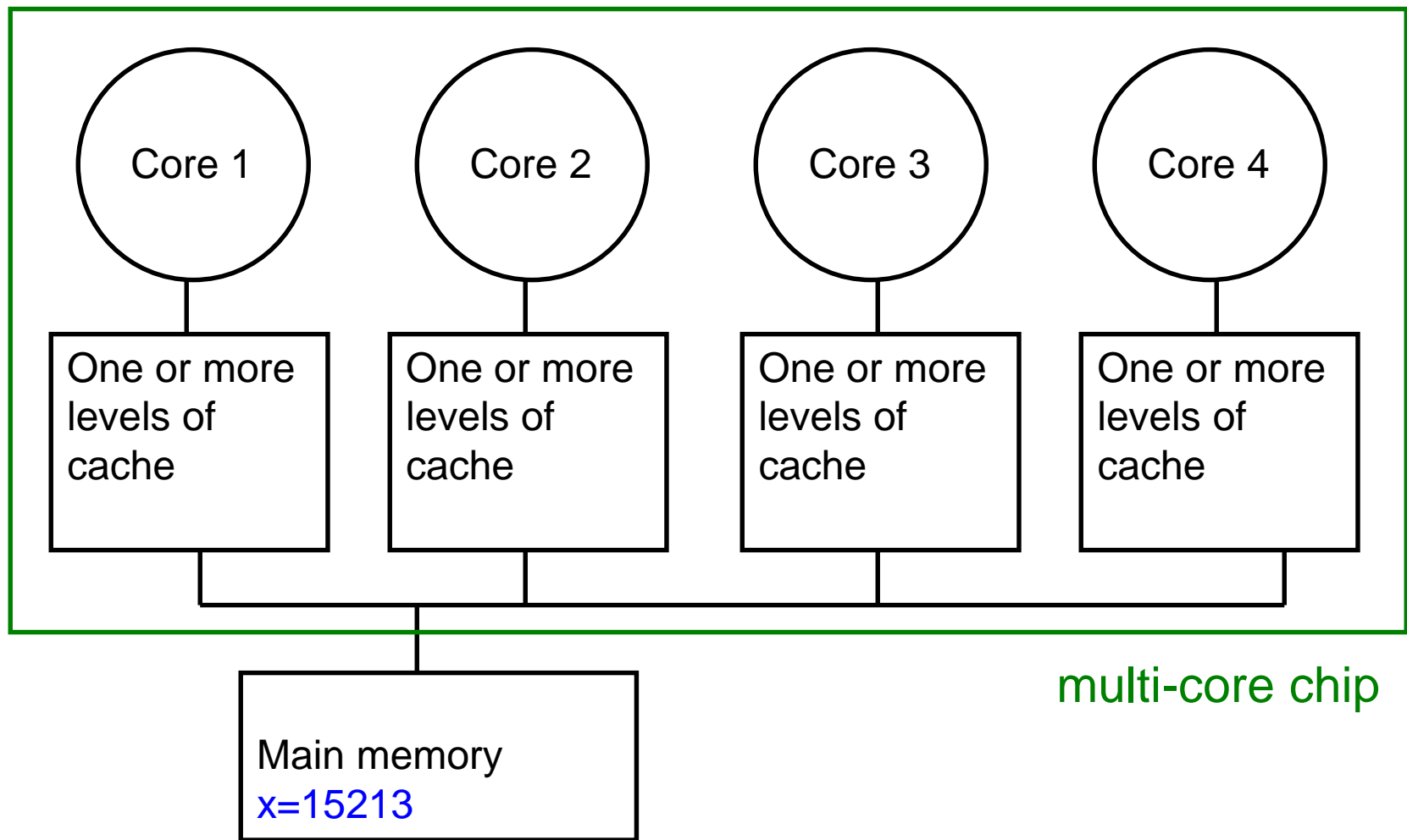
提交截止时间：2018年3月28日（周二） 22:00

提交方式：<http://home.ustc.edu.cn/~zzh04/pc.html>

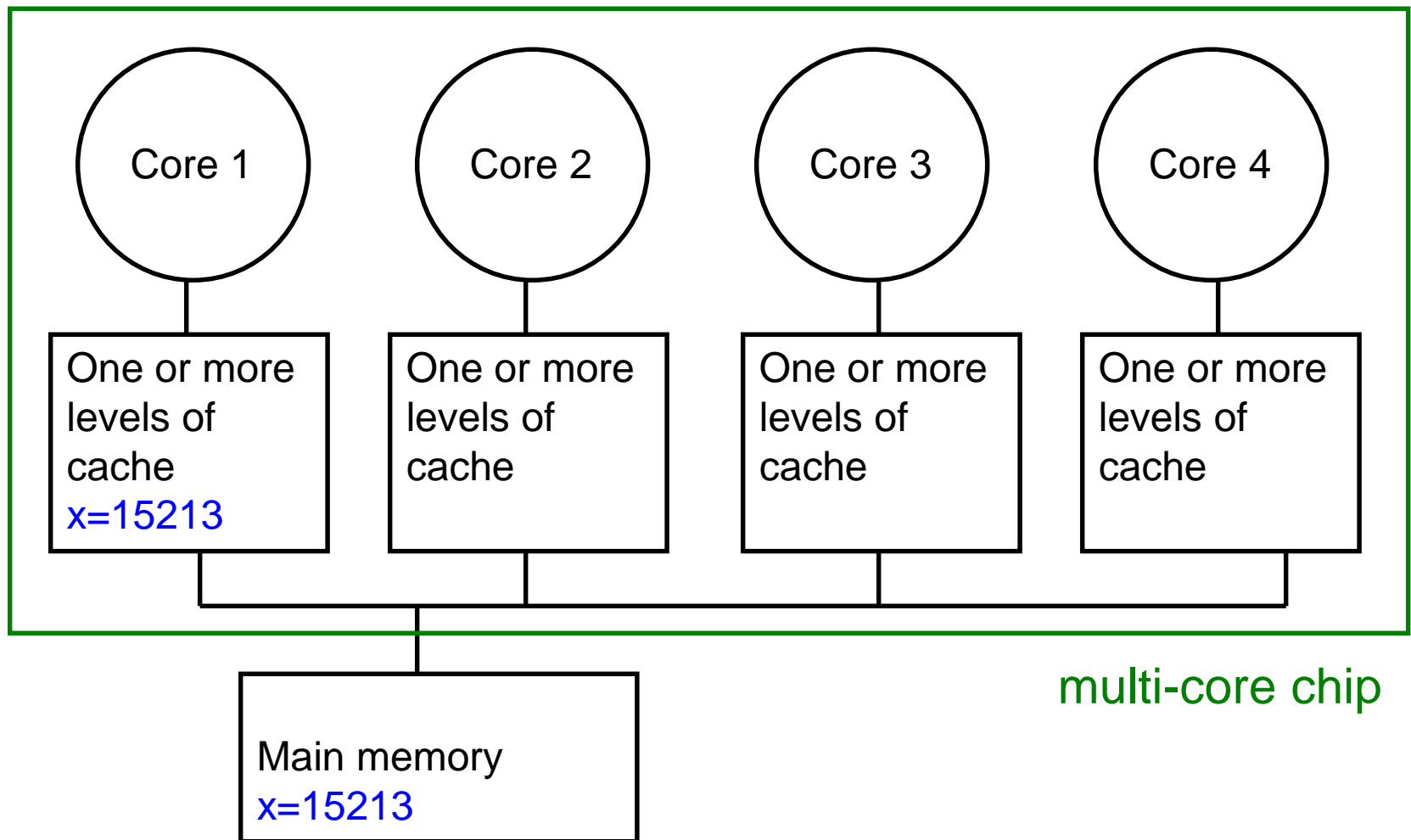
思考题2

- 与串行计算相比，并行机系统的存储结构更为复杂。试描述缓存一致性（cache coherence）问题。请考虑可以如何解决这个问题。

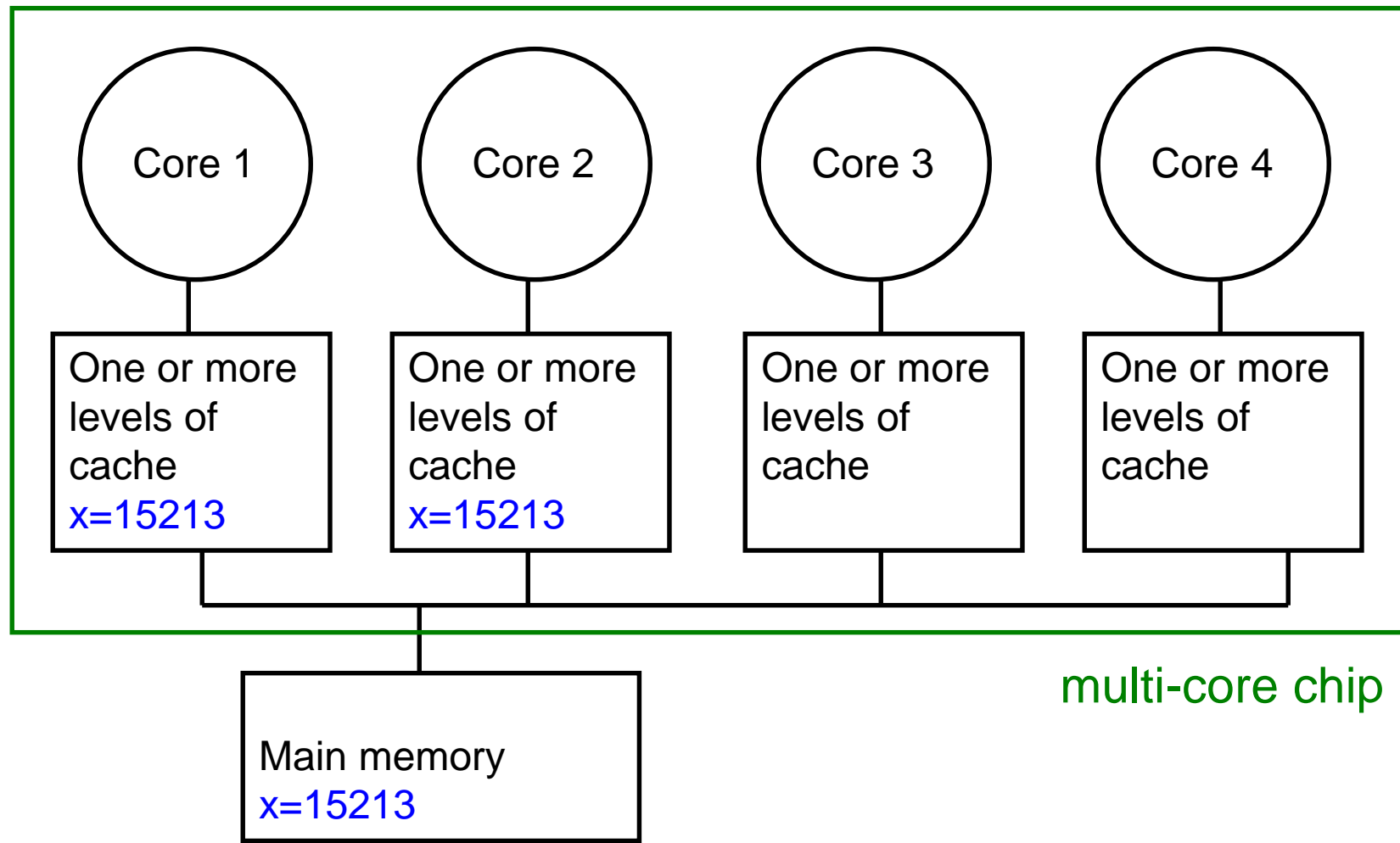




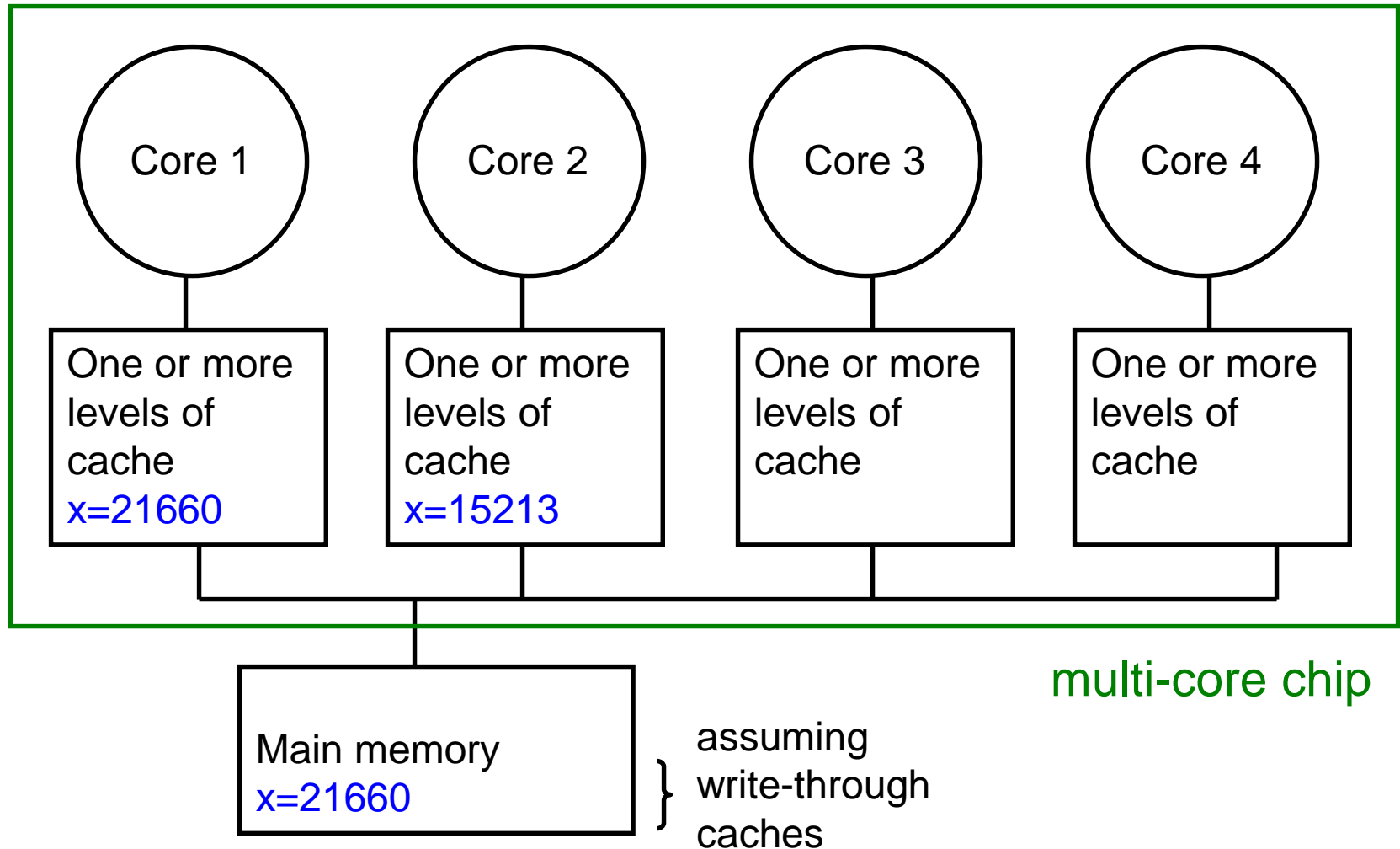
Core 1 reads x



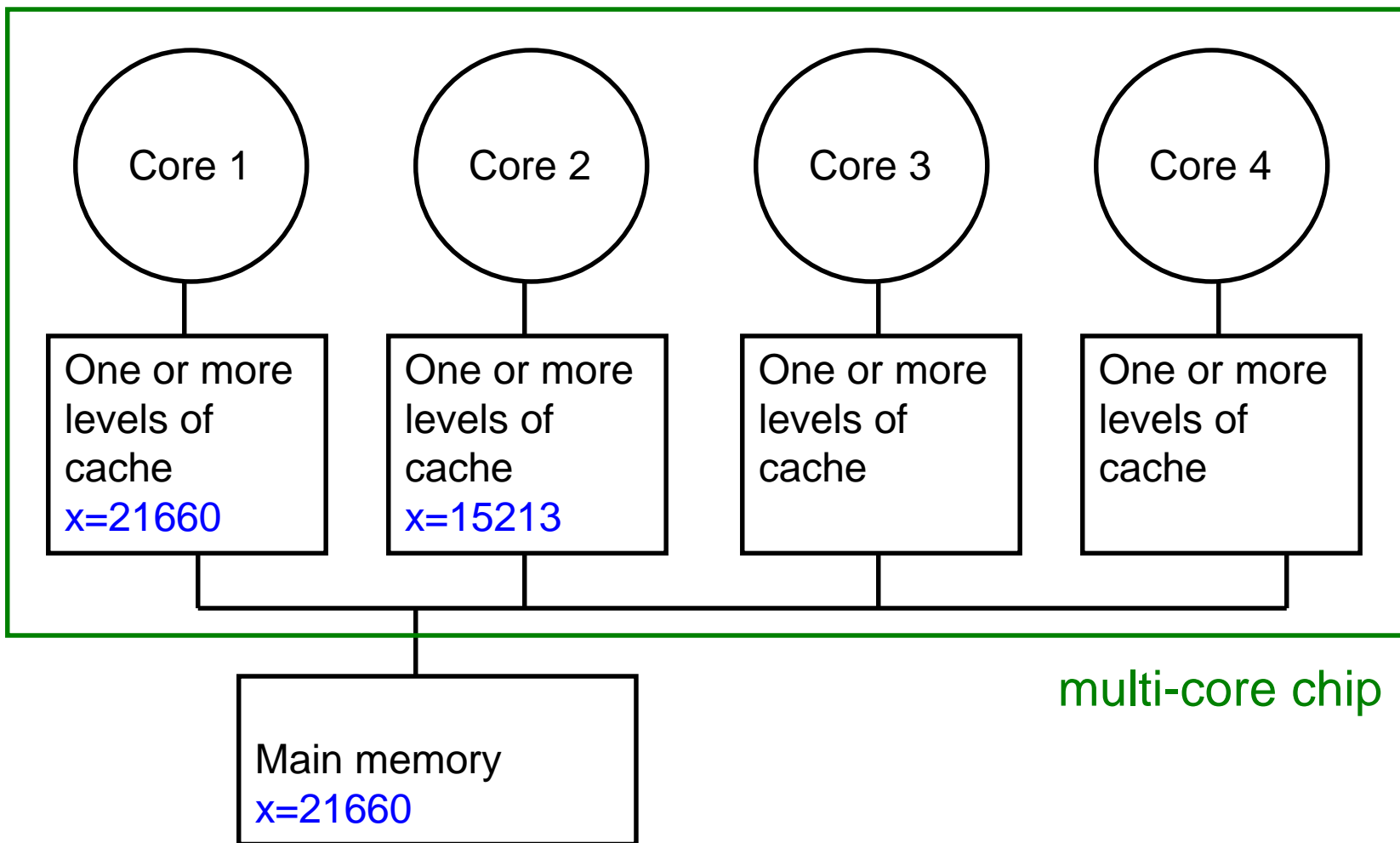
Core 2 reads x



Core 1 writes to x, setting it to 21660



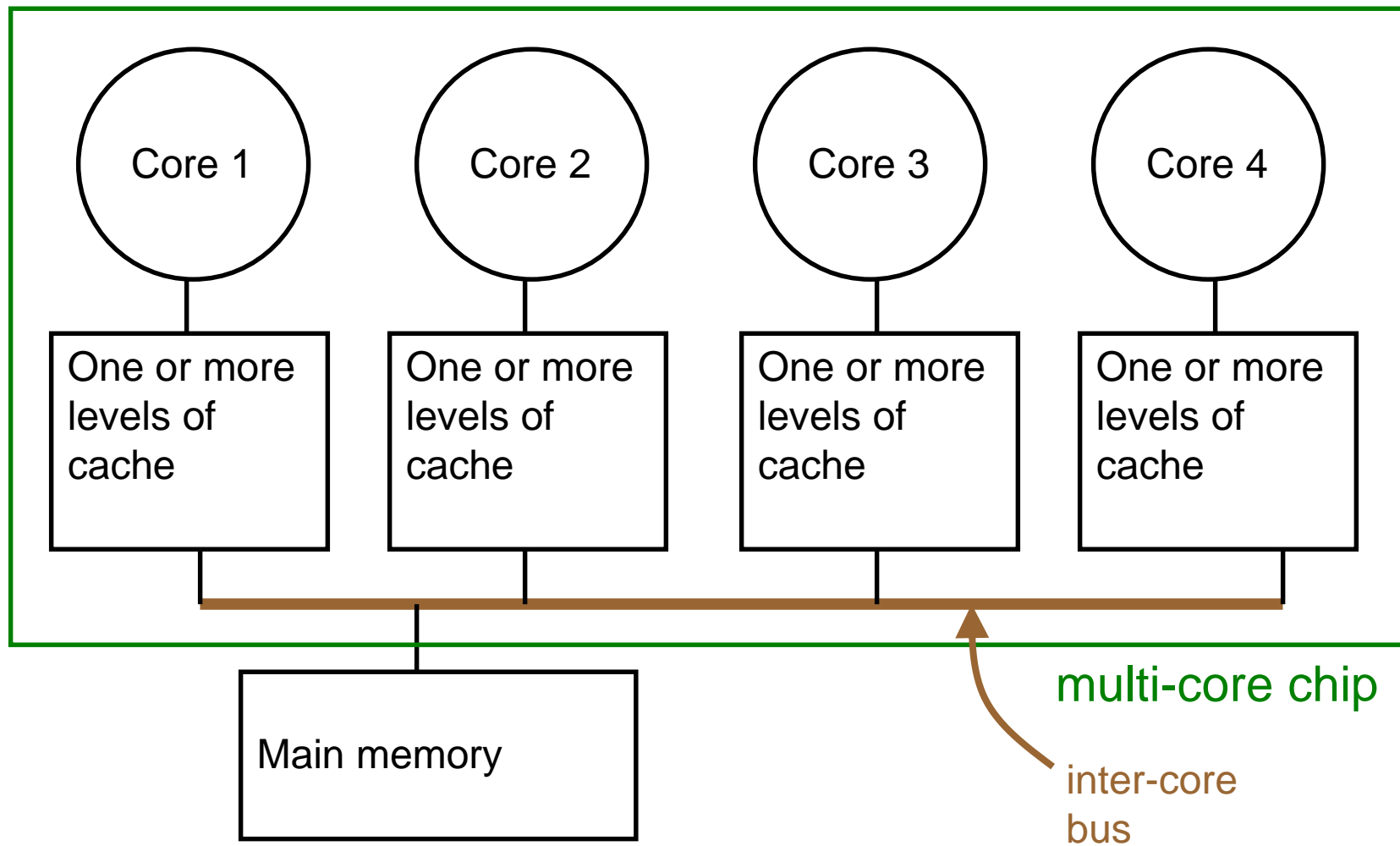
Core 2 attempts to read x... gets a stale copy



Solutions for cache coherence

- This is a general problem with multiprocessors, not limited just to multi-core
- There exist many solution algorithms, coherence protocols, etc.
- A simple solution:
invalidation-based protocol with *snooping*

Inter-core bus



Invalidation protocol with snooping

- Invalidation:

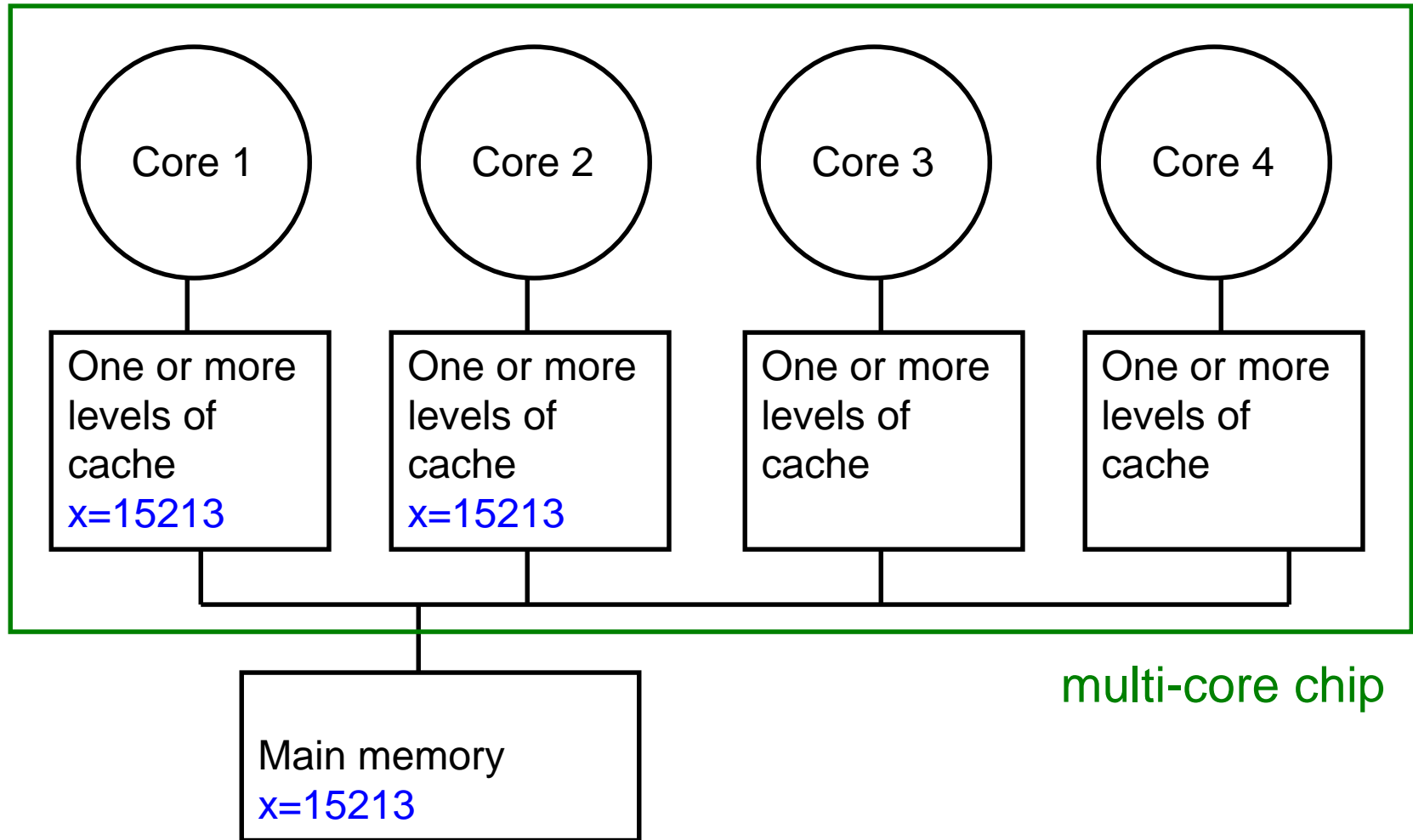
If a core writes to a data item, all other copies of this data item in other caches are *invalidated*

- Snooping:

All cores continuously “snoop” (monitor) the bus connecting the cores.

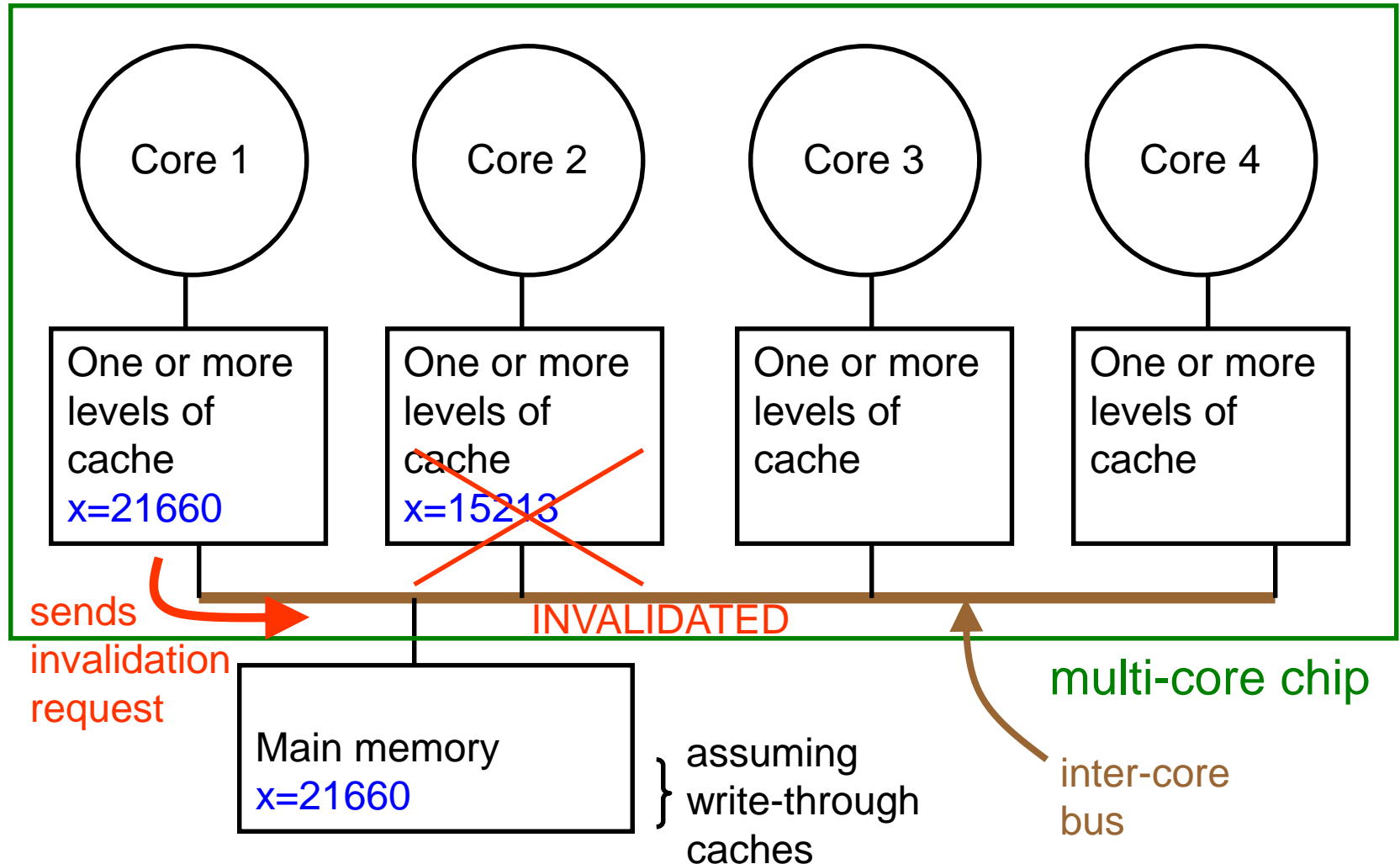
The cache coherence problem

Revisited: Cores 1 and 2 have both read x



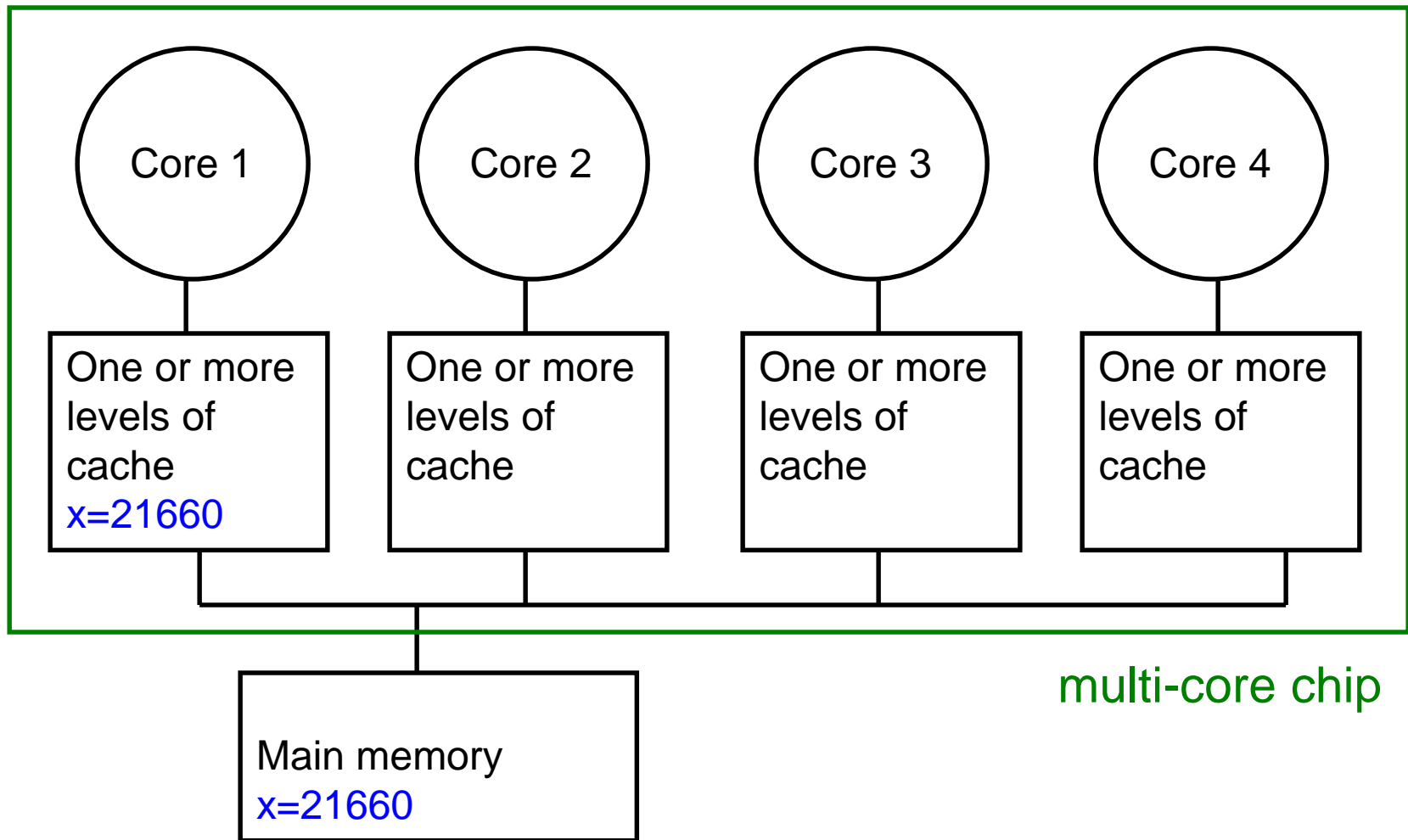
The cache coherence problem

Core 1 writes to x, setting it to 21660



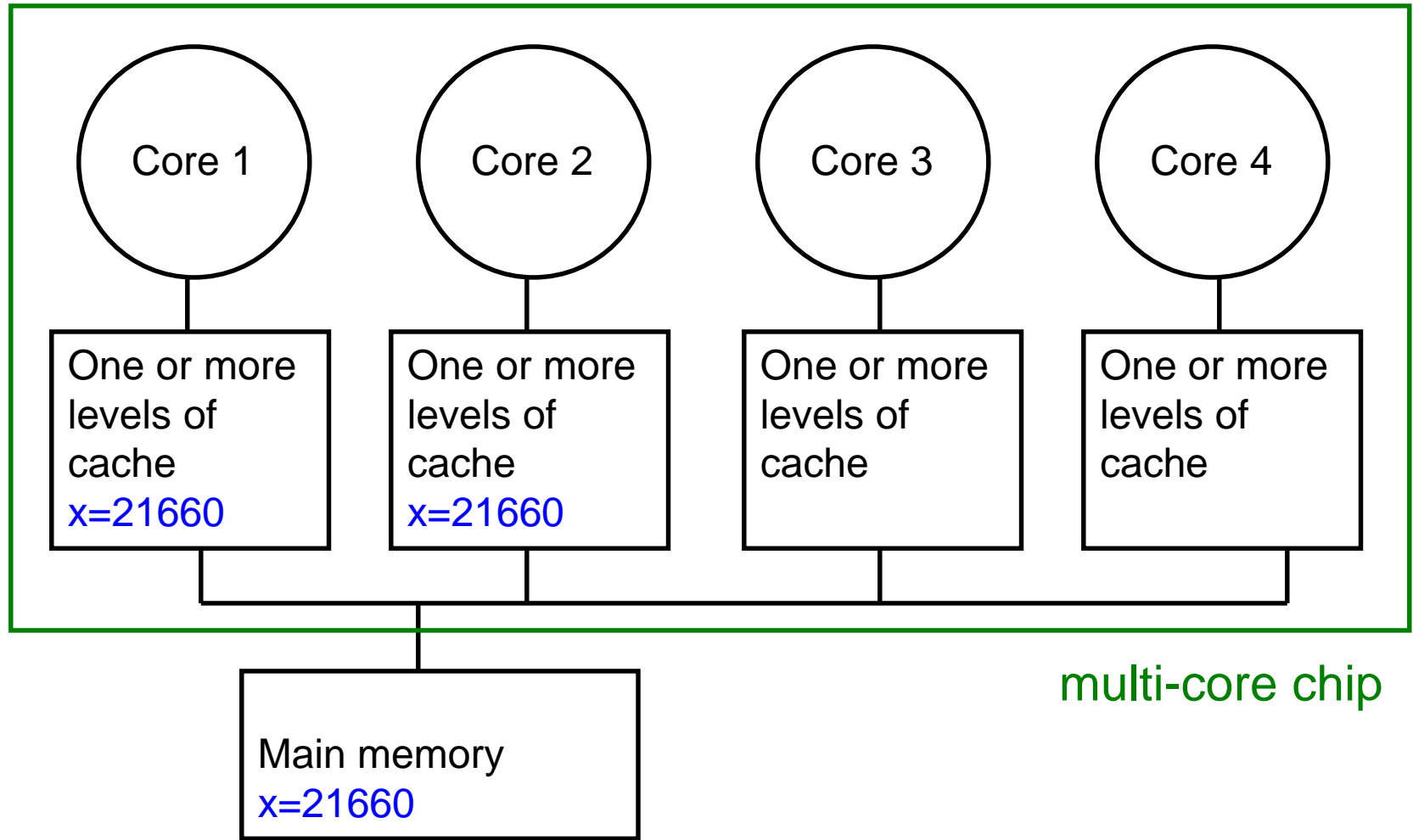
The cache coherence problem

After invalidation:



The cache coherence problem

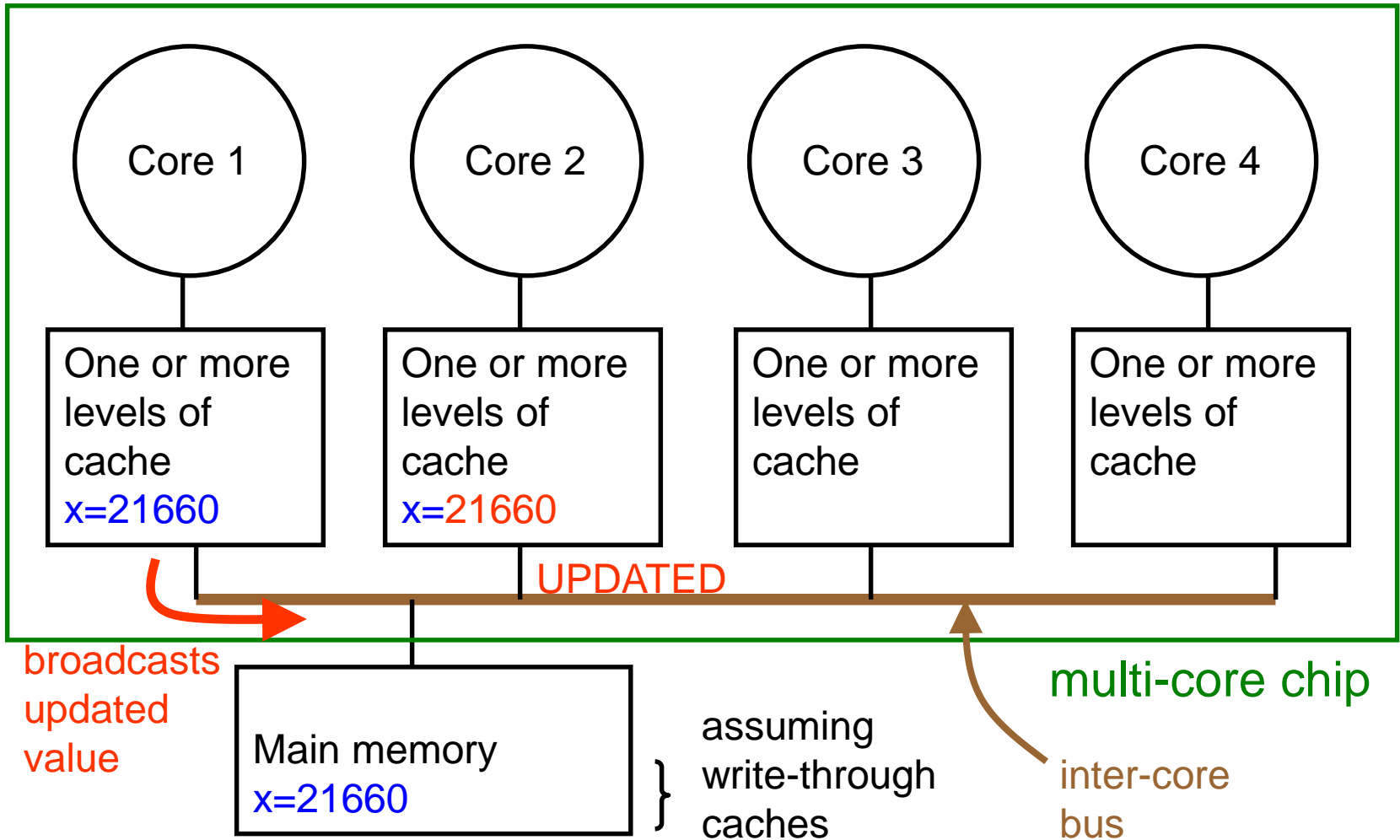
Core 2 reads x. Cache misses, and loads the new copy.



multi-core chip

Alternative to invalidate protocol: update protocol

Core 1 writes $x=21660$:



Which do you think is better?
Invalidation or update?

Invalidation vs update

- Multiple writes to the same location
 - invalidation: only the first time
 - update: must broadcast each write
(which includes new variable value)
- Invalidation generally performs better:
it generates less bus traffic