Cost of basic operations

2 billion cycles < 1 second

< 1 clock cycle: ***independent*** elementary arithmetic operations

- add, subtract, multiply, and, or, xor, shift; int, long long, float, double …

< 1 clock cycle: getting data from memory ***if easy to predict***

- linear reading

> 1 clock cycle: ***dependent*** elementary arithmetic operations

- `a += x1; a += x2; a += x3; a += x4; …`

> 10 clock cycles: ***non-elementary*** arithmetic operations

- division, square root, trigonometry

> 100 clock cycles: getting data from memory ***if hard to predict***

- following linked list

Dependencies

Independent operations — can do useful work all the time:

- `a1 += x1;  a2 += x2;  a3 += x3;  …`
- `a1 = v[i1]; a2 = v[i2]; a3 = v[i3]; …`

Dependent operations — lots of waiting:

- `a += x1;  a += x2;  a += x3;  …`
- `i2 = v[i1]; i3 = v[i2]; i4 = v[i3]; …`

Predictable memory accesses, independent operations:

- `for (int i = 0; i < n; ++i) { u[i] = v[i] + 123; }`

Hard-to-predict memory accesses, dependent operations:

- `while (i) { i = v[i]; }`

What can you expect to do in ≈ 1 second?

n =	1M	2M	4M	10M	20M	40M	100M	200M	400M	1G	2G
basic arithmetic: + - * | & ^ ~											x
linear reading											x
array updates (1K)										x	
division: / %									x		
array updates (1M)								x			
random number generator								x			
std::vector<bool>							x				
array updates						x					
follow linked list				x							
std::sort				x							
std::priority_queue<int>			x								
std::unordered_set<int>			x								
std::unordered_map<int,int>		x									
std::set<int>	x										
std::map<int,int>	x										