High Performance Computing and Communications Glossary Release 1.3a

K.A.Hawick
hawick@npac.syr.edu
Northeast Parallel Architectures Center
Syracuse University

24 July 1994

This is a **glossary** of terms on **High Performance Computing and Communications (HPCC)** and is essentially a **roadmap** or **information integration system** for HPCC technology. The master copy of this document is maintained online as HTML hypertext, where cross references are hypertext links to other entries in the document. Some entries, like that for ScaLAPACK software for example, point to the software source itself, available on the internet from sites like the National Software Exchange. In this "paper" version, entry headings are in **bold**, and cross reference entries are in *italics.*

The maintainance of this glossary is of necessity an ongoing process in an active field like HPCC, and so entries are being updated and augmented periodically. Check the version numbers. The master hypertext copy is stored at the *Northeast Parallel Architectures Centre (NPAC)* at URL:

http://www.npac.syr.edu/nse/hpccgloss

and NPAC's main URL is:

http://www.npac.syr.edu

An additional copy is also available in the Information Databases section of the *National Software Exchange* which can be accessed from the URL:

http://www.netlib.org/nse/home.html.

Several people have contributed to the entries in this glossary, especially **Jack Dongarra** and **Geoffrey Fox** and many of my colleagues at **Edinburgh** and **Syracuse**. Any further contributed entries, suggested revisions, or general comments will be gladly received at the email address above.

This paper document is also available by anonymous ftp (ftp.npac.syr.edu) as Technical Report SCCS-629, or by request from NPAC at 111 College Place, Syracuse NY 13244-4100, USA.

A

acyclic graph (n.) a graph without any cycles.
adaptive (adj.) Taking available information into account. For example, an adaptive mesh-generating algorithm generates a finer resolution mesh near to discontinuities such as boundaries and corners. An adaptive routing algorithm may send identical messages in different directions at different times depending on the local density information it has on message traffic. See also *oblivious*
address generation (n.) a cycle during execution of an instruction in which an effective address is calculated by means of indexing or indirect addressing.
address space (n.) A region of a computer's total memory within which addresses are contiguous and may refer to one another directly. A *shared memory* computer has only one user-visible address space; a *disjoint memory* computer can have several.
adjacency matrix (n.) a Boolean matrix indicating for each pair of vertices I and J whether there is an edge from I to J.
all-pairs shortest-path problem (n.) given a weighted graph, find the shortest path between every pair of vertices.
alpha-beta pruning (n.) an algorithm used to determine the *minimax* value of a game tree. Alpha-beta algorithm avoids searching subtrees whose evaluation cannot influence the outcome of the search.
Amdahl's Law (n.) A rule first formalised by Gene Amdahl in 1967, which states that if F is the fraction of a calculation that is serial and 1-F the fraction that can be parallelized, then the *speedup* that can be achieved using P processors is: 1/(F + (1-F)/P) which has a limiting value of 1/F for an infinite number of processors. This no matter how many processors are employed, if a calculation has a 10% serial component, the maximum speedup obtainable is 10.
AND tree (n.) a search tree whose nonterminal nodes are all AND nodes.
AND-parallelism (n.) A form of parallelism exploited by some implementations of parallel logic programming languages, in which the terms in conjunctions are evaluated simultaneously, and parent computations are allowed to proceed only once their immediate children have completed. See also *OR-parallelism* .
AND/OR tree (n.) a search tree with both AND and OR nonterminal nodes.
antidependence (n.) See *dependence*.
applicative language a language that performs all computations by applying functions to values.
architecture (n.) The basic plan along which a computer has been constructed. Popular parallel architectures include *processor arrays*, bus-based *multiprocessors* (with *caches* of various sizes and structures) and disjoint memory *multicomputers*. See also *Flynn's taxonomy*.
arity (n.) The number of arguments taken by a function. Arity is also sometimes used in place of *valence*.
ARPAnet (n.) The US Advanced Research Project Agency network, which was the first large multi-site computer network.

array constant (n.) an array reference within an iterative loop, all of whose subscripts are invariant.
array processor (n.) Any computer designed primarily to perform *data parallel* calculations on arrays or matrices. The two principle architectures used are the *processor array* and the *vector processor*.
artificial intelligence (n.) A class of problems such as pattern recognition, decision making and learning in which humans are clearly very proficient compared with current computers.
associative address (n.) a method whereby access is made to an item whose key matches an access key, as distinct from an access to an item at a specific address in memory. See *associative memory*.
associative memory (n.)Memory that can be accessed by content rather than by address; *content addressable* is often used synonymously. An associative memory permits its user to specify part of a pattern or key and retrieve the values associated with that pattern. The *tuple space* used to implement the *generative communication* model is an associative memory.
associative processor (n.) a *processor array* with *associative memory*.
asynchronous (adj.) Not guaranteed to enforce coincidence in *clock time*. In an asynchronous communication operation the sender and receiver may or may not both be engaged in the operation at the same instant in clock time. See also *synchronous*.
asynchronous algorithm (n.) See *relaxation method*.
asynchronous message passing (n.) See *message passing* .
atomic(adj.) Not interruptible. An atomic operation is one that always appears to have been executed as a unit.
attached vector-processor (n.) A specialised processor for vector computations, designed to be connected to a general purpose host processor. The host processor supplies *I/O* functions, a file system and other aspects of a computing system environment.
automatic vectorization (n.) A compiler that takes code written in a serial language (often Fortran or C) and translates it into vector instructions. The vector instructions may be machine specific or in a source form such as array extensions or as subroutine calls to a vector library. See also *compiler optimization*.
auxiliary memory (n.) Memory that is usually large, in capacity, but slow and inexpensive, often a rotating magnetic or optical memory, whose main function is to store large volumes of data and programs not being actively used by the processors.
AVL tree (n.) binary tree having the property that for any node in the tree, the difference in height between the left and right subtrees of that node is no more than 1.

B

back substitution (n.) See *LU decomposition*.
banded matrix (n.) a matrix in which the non-zero elements are clustered around the main diagonal. If all the non-zero elements in the matrix are within m columns of the main diagonal, then the *bandwidth* of the matrix is 2m+1.

bandwidth (n.) A measure of the speed of information transfer typically used to quantify the communication capability of *multicomputer* and *multiprocessor* systems. Bandwidth can express point to point or collective (*bus*) communications rates. Bandwidths are usually expressed in megabytes per second. Another usage of the term **bandwidth** is literally as the width of a band of elements in a matrix. See *banded matrices*.

bank conflict (n.) A bank "busy-wait" situation. Memory chip speeds are relatively slow when required to deliver a single word, so supercomputer memories are placed in a large number of independent banks (usually a power of 2). A vector of data laid out contiguously in memory with one component per successive bank, can be accessed at one word per cycle (despite the intrinsic slowness of the chips) through the use of pipelined delivery of vector-component words at high bandwidth. When the number of banks is a power of two, then vectors requiring strides of a power of 2 can run into a bank conflict.

bank cycle time (n.) The time, measured in clock cycles, taken by a memory bank between the honoring of one request to fetch or store a data item and accepting another such request. On most supercomputers this value is either four or eight clock cycles.

barrier (n.) A point in a program at which *barrier synchronization* occurs. See also *fuzzy barrier*.

barrier synchronization(n.) An event in which two or more processes belonging to some implicit or explicit group *block* until all members of the group have blocked. They may then all proceed. No member of the group may pass a *barrier* until all processes in the group have reached it. See also *fuzzy barrier*.

basic block (n.) A section of program that does not cross any conditional branches, loop boundaries or other transfers of control. Most *compiler optimization* is done within basic blocks.

batch search (n.) a concurrent search for a number of names.

benchmark (n.) a quantitative measure of performance for a computing system. The measure may be in terms of computational performance, which is often rated in *FLOPS*, or in terms of memory speed, or communications *bandwidth* or some more application-oriented measure such as *LIPS* or *OLTPS*. A collection of *benchmark results* for many computer systems is available on line from the *National Software Exchange*.

Bernstein's Condition (n.) A sufficient condition for the independence of two sections of a program as stated by Bernstein in 1966. If Ri(Wi) is the set of variables read(written) by a section of code i, then Bernstein's Condition states that sections i and j may be executed in an arbitrary order, or concurrently if: there is no *true dependence*, *output dependence* or *antidependence* among the statements in the sections.

BiCMOS (n.) Bi-polar *CMOS*. BiCMOS is a merger of *ECL* and CMOS wafer processes allowing both types of circuit to exist on the same chip. This gives the advantage of the small feature size and large scale integration of CMOS with ECL's high power, fast driver circuits.

binary semaphore (n.) a *semaphore* that can only take on the values 0 and 1.

bisection bandwidth(n.) The rate at which communication can take place between one half of a computer and the other. A low bisection bandwidth, or a large disparity between the maximum and minimum bisection bandwidths achieved by cutting the computers ele-

ments in different ways is a warning that communications bottlenecks may arise in some calculations.

bit-addressable (adj.) Allowing direct access to individual bits, rather than requiring bits to be selected by applying arithmetic or other operations to whole words. The local memory of each processing element in many *processor arrays* is bit addressable.

bitonic merge (n.)a parallel algorithm to merge two *bitonic sequences* of length 2^k into a single bitonic sequence of length 2^(k+1) in k+1 steps.

bitonic sequence (n.) a sequence of numbers A0,A1,...An-1 with the property that (1) there exists an index i, 0<=i<=n-1, such that A0 through Ai is monotonically increasing and Ai through An-1 is monotonically decreasing, or else (2) there exists a cyclic shift of indices so that condition 1 is satisfied.

BLAS (n.) Basic linear algebra software: a suite of very basic linear algebra routines, out of which almost any other matrix calculation can be built. See the *National Software Exchange* for the *BLAS* source and documentation.

block (v.) To suspend one's own operation, or the operation of another process. A process may block itself, or be blocked by the system, until some event occurs. If all processes are simultaneously blocked, and no external event can cause any of them to become unblocked, then *deadlock* has occurred. The term **block** is also often used to refer to a chunk of data or program. See also *basic block*.

blocking (adj.) An operation that causes the process executing it to block. Usually applied to communications operations, where it implies that the communicating process cannot perform any other operations until the communication has completed. See also *asynchronous*, *non-blocking* and *synchronous*.

boundary value swapping(n.) A technique used when performing calculations on a *mesh* on which *geometric decomposition* has been used. During a boundary value swap, each process exchanges values on the edges of its tile or tiles for the complementary values of its neighbours. See also *indirect method*.

BPS (n.) bytes per second, a unit of memory access speed or communications transfer speed. See also *WARPS* and *FLOPS*.

broadcast (n.) To send a message to all possible recipients. Broadcast can be implemented as a repeated send, but is more efficiently implemented by using *spanning trees* and having each node in the tree propagate the message to its descendants. See also *multicast* and *process group*.

buffer (n.) A temporary storage area in memory. Many methods for *routing* messages between processors use buffers at the source and destination, or at intermediate processors. See also *packet switching*, *virtual cut-through* and *wormhole routing*.

buffered message passing (n.) See *message passing system*.

bus (n.) A single physical communications medium shared by two or more devices. The *network* shared by processors in many *distributed computers* is a bus, as is the shared data path in many *multiprocessors*. See also *link*.

busy-waiting (n.) a situation whereby processor cycles are used to test a variable until it assumes a desired value.

butterfly(n.) A *topology* in which nodes are organised into levels, and there is a unique

path from any node in the first level to any node in the last. A butterfly is a recursive topology . See also *hypercube* and *shuffle exchange network*.

C

cache (n.) A high-speed memory, local to a single processor , whose data transfers are carried out automatically in hardware. Items are brought into a cache when they are referenced, while any changes to values in a cache are automatically written when they are no longer needed, when the cache becomes full, or when some other process attempts to access them. Also (v.) To bring something into a cache.
cache consistency (n.) The problem of ensuring that the values associated with a particular variable in the *caches* of several processors are never visibly different.
cache hit (n.) a cache access that successfully finds the requested data.
cache line (n.) The unit in which data is fetched from memory to cache.
cache miss (n.) A cache access that fails to find the requested data. The cache must then be filled from main memory at the expense of time.
CAD (n.) Computer-Aided Design; a term which can encompass all facets of the use of computers in manufacturing although the term *CAM* is also in use.
CAE (n.)Computer-Aided Engineering, like *CAD*, but usually applied to the use of computers in fields such as civil and nautical engineering.
CAM (n.) Computer-Aided Manufacturing.
cellular automata (n.) A system made up of many discrete cells, each of which may be in one of a finite number of states. A cell or automaton may change state only at fixed, regular intervals, and only in accordance with fixed rules that depend on cells own values and the values of neighbours within a certain proximity.
cellular computer (n.) A term sometimes used to describe fine grain systems such as *neural networks*, *systolic array*, and *SIMD* systems. Such systems tend to be well suited for the implementation of *cellular automata* .
CFD (n.) Computational fluid dynamics; the simulation or prediction of fluid flow using computers, a field which has generally required twice the computing power available at any given time.
chain (n.) A *topology* in which every processor is connected to two others, except for two end processors that are connected to only one other. See also *Hamiltonian*, *ring*.
chaining (n.) the ability to take the results of one vector operation and use them directly as input operands to a second vector instruction, without the need to store to memory or registers the results of the first vector operation. Chaining (or linking as it is sometimes called) can significantly speed up a calculation.
channel (n.) A point-to-point connection between two processes through which messages can be sent. Programming systems that rely on channels are sometimes called connection-oriented, to distinguish them from the more widespread connectionless systems in which messages are sent to named destinations rather than through named channels. See also *CSP*, *channel mask*.
channel mask (n.) A non-negative integer specifying a set of communication *channels*. If the numbers of the channels to be specified are I0,I1,...In-1, then the channel mask is the

integer for which these bit numbers are set to 1 and all other bits are 0. For example, if the channels to be specified are numbered 0, 1 and 4, the corresponding channel mask is the binary number 10011, or 19, in which the bits numbered 0, 1 and 4 are set to 1.

chime (n.) Chained vector time, approximately equal to the vector length in a DO-loop. The number of chimes required for a loop dominates the time required for execution. A new chime begins each time a resource such as a functional unit, vector register or memory path, must be reused.

chunksize (n.) The number of iterations of a parallel DO-loop grouped together as a single task in order to increase the granularity of the task.

CISC (adj.) Complicated instruction set computer; a computer that provides many powerful but complicated instructions. This term is also applied to software designs that give users a large number of of complex basic operations. See also *RISC*.

clausal logic (n.) a form of logic in which all propositions are expressed in terms of AND, OR and NOT. A six-stage process transforms any predicate calculus formula into clausal form. See also *clause*.

clause (n.) a sentence in formal logic. See also *clausal logic*.

clock cycle (n.) The fundamental period of time in a computer. Current technology will typically have this measured in nanoseconds.

clock time(n.) Physical or elapsed time, as seen by an external observer. Nonrelativistic time. In small computer systems where all components can be synchronized, clock time and *logical time*> may be the same everywhere, but in large systems it may be difficult for a processor to correlate the events it sees with the clock time an external observer would see. The clock times of events define a complete order on those events.

CMOS (n.) Complementary Metal Oxide on Silicon. A widely used chip technology. See also *BiCMOS*.

co-processor (n.) an additional processor attached to a main processor, to accelerate arithmetic, *I/O* or graphics operations.

coarse grain(adj.) See *granularity*

Cobegin/Coend (n.) a structured way of indicating a set of statements that can be executed in parallel.

combining (v.) Joining messages together as they traverse a *network*. Combining may be done to reduce the total traffic in the network, to reduce the number of times the start-up penalty of messaging is incurred, or to reduce the number of messages reaching a particular destination.

combining switch (n.) an element of an interconnection network that can combine certain types of requests into one request and produce a response that mimics serial execution of the requests.

common subexpression (n.) a combination of operations and operands that is repeated, especially in a loop. A good compiler will not recompute the common subexpressions but will save them in a register for reuse.

communicating sequential processes (n.) See *CSP*.

communication channel (n.) See *channel*

communication overhead (n.) A measure of the additional workload incurred in a parallel algorithm due to communication between the *nodes* of the parallel system. See also *latency*, *startup cost*

communication width (n.) the size of shared memory in an SIMD model assuming a global memory.

comparator (n.) a device that performs the *compare-exchange* operation.

compare-exchange the fundamental operation of the *bitonic merge* algorithm. Two numbers are brought together, compared, and then exchanged, if necessary, so that they are in the proper order.

compiler directives (n.) special keywords often specified as comments in the source code, but recognised by the compiler as providing additional information from the user for use in optimization.

compiler optimization (n.) Rearranging or eliminating sections of a program during compilation to achieve higher performance. Compiler optimization is usually applied only within *basic blocks* and must account for the possible *dependence* of one section of a program on another.

complex instruction set computer (adj.) See *CISC*.

complexity (n.) a measure of time or space used by an algorithm. Without adjective this refers to time complexity.

compress/index (n.) a vector operation used to deal with the nonzeroes of a large vector with relatively few nonzeroes. The location of the nonzeroes is indicated by an index vector (usually a bit vector of the same length in bits as the full vector in words). The compress operation uses the index vector to gather the nonzeroes into a dense vector where they are operated on with a vector instruction. See also *gather/scatter*.

computation-to-communication ratio (n.) The ratio of the number of calculations a process does to the total size of the messages it sends. A process that performs a few calculations and then sends a single short message may have the same computation-to-communication ratio as a process that performs millions of calculations and then sends many large messages. The ratio may also be measured by the ratio of the time spent calculating to the time spent communicating, in which case the ratio's value depends on the relative speeds of the processor and communications medium, and on the *startup cost* and *latency* of communication. See also *granularity*.

concurrent computer (n.) A generic category, often used synonymously with *parallel computer* to include both *multicomputer* and *multiprocessor*.

concurrent processing (adj.) simultaneous execution of instructions by two or more processors within a computer.

condition synchronization (n.) process of delaying the continued execution of a process until some data object it shares with another process is in an appropriate state.

configuration (n.) A particular selection of the types of processes that could make up a parallel program. Configuration is trivial in the *SPMD* model, in which every processor runs a single identical process, but can be complicated in the general *MIMD* case, particularly if user-level processes rely on libraries that may themselves require extra processes. See also *mapping*.

conjugate gradient method (n.) A technique for solving systems of linear algebraic equations, which proceeds by minimizing a quadratic residual error function. The method is iterative but quite powerful: in the absence of roundoff error, it will converge exactly in M steps, where M is the order of the system in question.
content addressable (adj.) See *associative memory*.
contention (n.) Conflict that arises when two or more requests are made concurrently for a resource that cannot be shared. Processes running on a single processor may contend for CPU time, or a *network* may suffer from contention if several messages attempt to traverse the same *link* at the same time.
context switching (n.) Saving the state of one process and replacing it with that of another that is *time sharing* the same processor. If little time is required to switch contexts, *processor overloading* can be an effective way to hide *latency* in a *message passing system*.
control process (n.) A process which controls the execution of a program on a *concurrent computer*. The major tasks performed by the control process are to initiate execution of the necessary code on each *node* and to provide *I/O* and other service facilities for the nodes.
control-driven (adj.) refers to an *architecture* with one or more program counters that determine the order in which instructions are executed.
cost (n.) complexity of an algorithm multiplied by the number of processors used.
cpu (n.) central processing unit or processor unit of a sequential computer system. Sometimes used to mean one processor element of a *concurrent computer* system.
critical node (n.) when a node is inserted into an AVL tree, the critical node is the root of the subtree about which a rebalancing is going to take place.
critical section (n.) a section of program that can be executed by at most one process at a time.
CSP (n.) Communicating sequential processes; an approach to parallelism in which anonymous processes communicate by sending messages through named point-to-point *channels*. CSP was coined by Hoare in 1985. All communication is *synchronous* in that the process that reaches the communication operation first is *blocked* until a complementary process reaches the same operation. See also *guard*.
cube-connected cycles network (n.) a processor organization that is a variant of a *hypercube*. Each hypercube node becomes a cycle of nodes, and no node has more than three connections to other nodes.
cycle a cycle of the computer clock is an electronic signal that counts a single unit of time within a computer.
cycle time (n.) the length of of a single cycle of a computer function such as a memory cycle or processor cycle. See also *clock cycle*.
cyclic reduction (n.) a parallel algorithm to solve general first order linear recurrence relations.

D

data cache (n.) a cache that holds data but does not hold instructions.
data dependency (n.) a situation existing between two statements if one statement can store into a location that is later accessed by the other statement. See also *dependence*.

data flow analysis (n.) process of finding dependencies among instructions.
data flow graph (n.) (1) machine language for a data flow computer; (2) result of data flow analysis.
data parallelism (n.) A model of parallel computing in which a single operation can be applied to all elements of a data structure simultaneously. Typically, these data structures are arrays, and the operations are arithmetic and act independently on every array element, or *reduction operations*. See also *array processor*, *processor array*, *SIMD* and *vector processor*.
data-driven (n.) a data flow architecture in which execution of instructions depends on availability of operands.
dataflow (n.) A model of parallel computing in which programs are represented as *dependence graphs* and each operation is automatically *blocked* until the values on which it depends are available. The parallel functional and parallel logic programming models are very similar to the dataflow model.
dead code(n.) A portion of a program that does not have to be executed (because the values it calculates will never be used) or that will never be entered. *Compiler optimization* usually removes sections of dead code. See also *dependence*.
deadlock (n.) A situation in which each possible activity is *blocked*, waiting on some other activity that is also blocked. If a directed graph represents how activities depend on others, then deadlock arises if and only if there is a cycle in this graph. See also *dependence graph*.
decision problem (n.) a problem whose solution, if any, is found by satisfying a set of constraints.
declustered (adj.) A *file system* that distributes blocks of individual files between several disks. This contrasts with a traditional file system, in which all blocks of a single file are placed on the same disk. See also *striped*.
decomposition (n.) A division of a data structure into substructures that can be distributed separately, or a technique for dividing a computation into subcomputations that can be executed separately. The most common decomposition strategies in parallel computing are:

- functional decomposition involves breaking a calculation up into qualitatively different subcalculations, such as the transformation, shading, z-buffering and rendering portions of a polygon display algorithm.

- geometric decomposition involves breaking a calculation into sections that correspond to some physical subdivision of the system being modeled, such as *tiling* a mesh. To achieve good *load balance* such sections may be either distributed according to some regular rule (regular domain decomposition) or scattered randomly (scattered spatial decomposition).

- iterative decomposition involves breaking down a calculation in which one or more operations are repeatedly applied to one or more data values by treating each data subset / operation subset as a separate task and distributing tasks amongst available processors, either deterministically or randomly. In a deterministic decomposition, the data to be processed are fixed and the same operations are applied to each. In

a speculative decomposition, different operations are applied simultaneously to the same input until at least one has completed.

dedicated throughput (n.) the number of results returned for a single job per time unit.
demand-driven (n.) *data flowarchitecture* in which execution of an instruction depends upon both availability of its operands and a request for the result.
dependence (n.) The relationship of a calculation B to a calculation A if changes to A, or to the ordering of A and B, could affect B. If A and B are calculations in a program, for example, then B is dependent on A if B uses values calculated by A. There are four types of dependence:

- true dependence: B uses values calculated by A.
- antidependence: A uses values overwritten by B.
- output dependence: A and B both write to the same variables.
- control dependence: B's execution is controlled by values set in A.

Dependence is also used in message *routing* to mean that some activity X cannot proceed until another activity Y has completed. For example, if X and Y are messages attempting to pass through a region with limited *buffer* space, and Y currently holds some or all of the buffer, X may depend on Y releasing some buffer space before proceeding.
dependence graph (n.) A directed graph whose nodes represent calculations and whose edges represent dependencies among those calculations. If the calculation represented by node k depends on the calculations represented by nodes i and j, then the dependence graph contains the edges i-k and j-k. See also *compiler optimization*, *dataflow*, *dependence*.
dependence analysis (n.) an analysis by compiler or precompiler that reveals which portions of a program depend on the prior completion of other portions of the program. Dependency analysis usually relates statements in an iterative code construct.
depth (n.) parallel time complexity.
deque (n.) a double ended queue; that is a list of elements on which insertions and deletions can be performed at both the front and rear.
deterministic model (n.) a task model in which precedence relations between tasks and the execution time needed by each task are fixed and known before the schedule is devised.
diameter (n.) The distance across a graph, measured by the number of *links* traversed. Diameter is usually taken to mean maximum diameter (ie the greatest internode distance in the graph, but it can also mean the average of all internode distances. Diameter is sometimes used as a measure of the goodness of a *topology*.
direct mapping (n.) a cache that has a set associativity of one so that each item has a unique place in the cache at which it can be stored.
direct memory access (n.) See *DMA*.
direct method (n.) Any technique for solving a system of equations that relies on linear algebra directly. LU decomposition with back substitution is an example of a direct method. See also *indirect method*.

direct naming (n.) a *message passing scheme* in which source and destination designators are the names of processes.
directed graph (n.) a graph in which the edges have an orientation, denoted by arrowheads.
disjoint memory (n.) Memory that appears to the user to be divided amongst many separate *address spaces*. In a *multicomputer*, each processor typically has its own *private memory* and manages requests to it from processes running on other processors. Disjoint memory is more commonly called *distributed memory*, but the memory of many *shared memory*> computers is physically distributed.
disk striping (n.) technique of interleaving a disk file across two or more disk drives to enhance input/output performance. The performance gain is a function of the number of drives and channels used.
distributed computer (n.) A computer made up of many smaller and potentially independent computers, such as a *network* of workstations. This *architecture* is increasingly studied because of its cost effectiveness and flexibility. Distributed computers are often *heterogeneous*. See also *multi-processor*, *multicomputer*.
distributed memory (n.) Memory that is physically distributed amongst several modules. A *distributed memoryarchitecture* may appear to users to have a single *address space* and a single *shared memory* or may appear as *disjoint memory* made up of many separate *address spaces*.
divide and conquer (n.) a problem solving methodology that involves partitioning a problem into subproblems, solving the subproblems, and then combining the solutions to the subproblems into a solution for the original problem.
DMA (n.) Direct Memory Access; allows devices on a bus to access memory without requiring intervention by the *CPU*.
domain (n.) That part of a larger computing resource allocated for the sole use of a specific user or group of users. See also *space sharing*.
domain decomposition (n.) See *decomposition*
DRAM (n.) Dynamic *RAM*; memory which periodically needs refreshing, and is therefore usually slower than *SRAM* but is cheaper to produce.
dusty deck (n.) A term applied to old programs (usually Fortran or Cobol). The term is derived from the image of a deck of punched cards grown dusty over the years.
dynamic channel naming (n.) a *message passing scheme* allowing source and destination designators to be established at run time.
dynamic decomposition (n.) a task allocation policy that assumes tasks are generated at execution time. See also *decomposition*.

E

e-cube routing(n.) A message *routing* algorithm used on binary *hypercubes*. If <—Sn-1...S0—> and <—Dn-1...D0—> are the binary co-ordinates of the source and destination of the message, and <—Xn-1...X0>— their difference, then the e-cube algorithm routes

the message along *link* parallel to the axes corresponding to 1's in <—Xn-1...X0—>, in order from highest to lowest. See also *Metropolis routing*, *randomized routing*.

eager evaluation (n.) A scheduling policy under which each computation begins as soon as its inputs are ready or its necessary preconditions have been satisfied. This is the scheduling policy used in most programs, but contrasts with the *lazy evaluation* policy often used in functional and logic programming. See also *dataflow*, *dependencedependence graph*.

ECL (n.) Emitter-coupled logic; a high speed, high power transistor technology for chip manufacture. See also *BiCMOS*, *CMOS*.

efficiency (n.) A measure of hardware utilization, equal to the ratio of *speedup* achieved on P processors to P itself. See also *isoefficiency*, *optimality*.

enumeration sort (n.) a sort that finds the position of each name by determining the number of names smaller than it.

EPROM (n.) Electronically programmable *ROM*; a memory whose contents can be changed using special hardware. This usually involves removing the chips from their environment in order to "burn" a new pattern into them.

ethernet (n.) a popular *LAN* technology invented by Xerox. See also *FDDI*.

expand(n.) a vector computer instruction that stores the elements of a vector according to the values in a corresponding masking vector.

expected space complexity (n.) the average amount of space used by an algorithm over all possible inputs.

expected time complexity (n.) the average amount of time used by an algorithm over all possible inputs.

explicitly parallel (n.) language semantics that describe which computations are independent and can be performed in parallel. See also *implicitly parallel*.

F

fact (n.) in the context of logic programming, a fact is a *Horn clause* with a head but no body.

fairness (n.) A property of a concurrent system. If a system is fair, then in the long run all processes are served equally. No process has preferential access to *semaphores* and in particular no process can *livelock*. See also *deadlock*.

fast Fourier transform (n.) See *FFT*

FDDI (n.) Fast digital data interface; a standard for fibre optic communications systems. See also *ethernet*.

fetch-and-add (n.) a computer synchronization instruction that updates a word in memory, returns the value before the update, and produces a set of results as if the processors executed in some arbitrary order.

FFT (n.) The fast Fourier transform is a technique for the rapid calculation of discrete Fourier transform of a function specified discretely at regular intervals. The technique makes use of a *butterfly* data structure.

FIFO (n.) First in, first out, a queue.

file system (n.) The hardware used for nonvolatile data storage; the system software that controls this hardware; the *architecture* of this hardware and software. A parallel file system is one that can be read or written to by many processors simultaneously. See also *RAID*.

fine grain (adj.) See *granularity*

finite difference method (n.) A *direct method* for the approximate solution of partial differential equations on a discrete grid, by approximating derivatives of the unknown quantities on the grid by linear differences. *Array processors* lend themselves very well to the efficient implementation to this sort of application. See also *finite element method*.

finite element method (n.) An approximate method for solving partial differential equations by replacing continuous functions by piecewise approximations defined on polygons, which are referred to as elements. Usually polynomial approximations are used. The finite element method reduces the problem of finding the solution at the vertices of the polygons to that of solving a set of linear equations. This task may then be accomplished by a number of methods, including *Gaussian elimination*, the *conjugate gradient method* and the *multigrid method*. See also *finite difference method*.

flit (n.) A flow control unit - the basic unit of information sent through a *message passing* system that uses *virtual cut-through* or *wormhole routing*.

FLOPS (n.) Floating point operations per second; a measure of memory access performance, equal to the rat eat which a machine can perform single-precision floating-point calculations. See also *WARPS*.

Flynn's Taxonomy (n.) A classification system for *architectures* that has two axes: the number of instructions streams executing concurrently, and the number of data sets to which those instructions are being applied. The scheme was proposed by Flynn in 1966. See also *SPMD*, *MIMD*, *SIMD*, *SISD*.

forall (n.) A programming construct that specifies a set of loop iterations and further specifies that these iterations can be done in any order. *data parallel* and *shared variable* programs are often expressed using forall loops.

fork (v.) To create a new process that is a copy of its immediate parent. See also: *join*, *spawn>*.

forward reduction (n.) See *LU decomposition*.

FPU (n.) Floating point unit; either a separate chip or an area of silicon on the *CPU* specialised to accelerate floating point arithmetic.

full matrix (n.) A full matrix is one in which the number of zero elements is small (in some sense) compared to the total number of elements. See also *sparse matrix*.

functional decomposition (n.) See *decomposition*.

functional unit (n.) functionally independent part of the ALU each of which performs a specific function, for example: address calculation; floating-point add or floating-point multiply.

futures (n.) A programming construct specifying that the value of some computation will be needed at some later point, but allowing the system to schedule that computation to run at any arbitrary time. Futures are one way to present *lazy evaluation* in *shared variable* programming. See also *eager evaluation*.

fuzzy barrier(n.) A pair of points in a program such that no process may pass the second point before all processes have reached the first. Fuzzy barriers are often used inside loop iterations to ensure that no changes are made during an iteration before all values needed in the previous iteration have been read. See also *barrier*, *barrier synchronization*, *dependence*.

G

GaAs (n.) Gallium arsenide; an relatively new semiconductor material, still not yet fully mastered by chip manufacturers. GaAs components can run much faster than silicon-based components. See also *CMOS*.
game tree (n.) the state space tree representation of a game position.
Gantt chart (n.) a diagram used to illustrate a deterministic schedule.
gather/scatter (n.) the operations related to large sparse data structures. A full vector with relatively few nonzeroes is transformed into a vector with only those nonzeroes by using a gather operation. The full vector, or one with the same structure, is built from the inverse operation or scatter. The process is accomplished with an index vector, which is usually the length of the number of nonzeroes, with each component being the relative location in the full vector. See also *compress/index*.
Gauss-Seidel method (n.) An iterative method for solving partial differential equations on a grid. When updating a grid point the new value depends on the current values at the neighbouring grid points, some of which are from the previous iteration and some, which have already been updated, are from the current iteration. So updates are performed by sweeping through the grid in some user-chosen fashion. The key feature of this algorithm is that it makes use of new information (updated grid point values) as soon as they become available.
Gaussian elimination (n.) A method for solving sets of simultaneous linear equations by eliminating variables from the successive equations. The original equation in the form Ax=b (A is a matrix, b the vector of known values, and x the unknown solution vector) is reduced to Ux=c, where U is an upper triangular matrix. The solution vector x can then be found by *back substitution*. This method is usually formulated as *LU decomposition*.
GAXPY (n.) a generalised *SAXPY* operation, taking linear combinations of a set of columns and accumulating the columns in a vector, as in a matrix-vector product.
generative communication(n.) A model of parallel computing in which processes that have been *spawned* dynamically turn into data upon completion, and data may be stored in *tuples* in one or more shared *tuple spaces*. A process may add tuples to a tuple space, or remove then by matching against their contents. See also *associative memory*, *shared variables*, *virtual shared memory*.
geometric decomposition (n.) See *decomposition*.
gigaFLOPS (n.) 10^9 *FLOPS*
GKS (n.) the Graphics Kernel Standard; a graphics standard developed for pen plotters and now supported on a wide variety of pixel based devices.
grain (n.) See *granularity*
granularity (n.) The size of operations done by a process between communications events. A fine grained process may perform only a few arithmetic operations between processing

one message and the next, whereas a coarse grained process may perform millions. See also *computation-to-communication ratio*.

graph (n.) an entity consisting of a set of vertices and a set of edges between pairs of vertices.

Gray code (n.) A *mapping* which labels the lattice points on an n-dimensional grid with the integers 0,1,2,...2^d-1, so that the labels at adjacent grid points differ in precisely one bit in their integer representation.

Grosch's Law (n.) an empirical rule that the cost of computer systems increases as the square root of the computational power of the systems.

guard (n.) A logical condition that controls whether a communication operation can take place. Guards are usually defined as part of the syntax and semantics of *CSP*-based langauges.

guard ring (n.) Parallel algorithms operating on a two dimensional grid generally store grid values in a two dimensional array, *geometrically decomposed* across the *nodes* of the *parallel computer*. The guard ring is the additional storage allocated around the edges of the array to hold the values at the grid points lying along the adjacent boundaries of neighbouring nodes. These values are obtained by communication between nodes.

Gustafson's Law(n.) A rule stating that if the size of most problems is scaled up sufficiently, then any required *efficiency* can be achieved on any number of processors, as stated by Gustafson in 1988. See also *Amdahl's Law*, *efficiency*.

H

halo (n.) The region around a point in a *mesh*, whose values are used when updating the value of the central point. In a *cellular automata*, the halo comprises those neighbouring cells whose values are used by the automaton's update rule.

Hamiltonian(n.) A closed path through a graph which passes through each node in the graph exactly once.

Hawick (n.) a Scots word meaning a "town surrounded by a hedge"; also an actual town on the border between Scotland and England; also my surname. This is not relevant to *HPCC* except that this is a usful way of ensuring my email address (**hawick@npac.syr.edu**) does not get lost from this file so you can always seek out the latest version of this glossary.

height (n.) in graph theory, height is the length of the longest path from the root of a tree to one of its leaves.

heterogeneous(adj.) Containing components of more than one kind. A heterogeneous *architecture* may be one in which some components are processors, and others memories, or it may be one that uses different types of processor together. See also *distributed computer*, *homogeneous*.

high-order interleaving (n.) memory interleaving strategy based on high-order bits of an address.

HiPPI (n.) High performance parallel interface; a point to point 100 MByte/sec interface standard used for networking components of high performance *multicomputers* together.

hit ratio (n.) the ratio of the number of times data requested from a cache is found (or hit) to the number of times it is not found (or missed).

homogeneous(adj.) Made up of identical components. A homogeneous *architecture* is one in which each element is of the same type; *processor arrays* and *multicomputers* are usually homogeneous. See also *heterogeneous*.
horizontal processing (n.) act of processing a two dimensional array row by row.
Horn clause (n.) a clause that contains at most one conclusion.
hot-spot contention (n.) an interference phenomenon observed in multiprocessors caused by memory access statistics being slightly skewed from a uniform distribution to favour a specific memory module.
HPCC (n.) an acronymn for High Performance Computing and Communications, which is the field of information addressed by this glossary. A *USA National Coordination Office* for HPCC also exists, and other information on HPCC can be found from the *Northeast Parallel Architectures Center*, the *Center for Research in Parallel Computing* the *National Software Exchange* or the *Edinburgh Parallel Computing Centre* depending upon your geography.
hypercube (n.) A *topology* of which each node is the vertex of a d-Dimensional cube. In a binary hypercube, each *node* is connected to n others, and its co-ordinates are one of the 2^n different n-bit sequences of binary digits. Most early American *multicomputers* used hypercubic topologies, and so the term hypercube is sometimes used as a synonym for multicomputers. See also *butterfly*, *e-cube routing*, *shuffle exchange network*.

I

I/O (n.) refers to the hardware and software mechanisms connecting a computer with its environment. This includes connections between the computer and its disk and bulk storage system and also connections to user terminals, graphics systems, and networks to other computer systems or devices. Standard I/O is a particular software package developed under UNIX for the C programming language.
implicitly parallel (n.) language semantics that do not allow the user to explicitly describe which computations are independent and can be performed in parallel. For an implicitly parallel language, the compiler must deduce or prove independence in order to make use of parallel hardware. The comparative difficulty of the deduction separates implicitly parallel languages from explicitly parallel ones.
indirect method (n.) Any technique for solving a system of equations that does not rely on linear algebra directly. Successive over-relaxation is an example of an indirect method. See also *direct method*.
information (n.) a collection of related data objects.
inner product method (n.) method of matrix multiplication in which one element of the resultant matrix is computed at a time. See also *middle product method*
input/output (n.) See *I/O*.
instruction buffering (n.) process of prefetching instructions with the goal of never making the processor wait for an instruction to be fetched. This is sometimes also known as instruction look-ahead.
instruction cache (n.) a cache memory that holds only instructions but not data.

instruction pipelining (n.) strategy of allowing more than one instruction to be in some stage of execution at the same time. See also *MISD*.
instruction scheduling (n.) a strategy of a compiler to analyse the outcome of the operations specified in a program and to issue instructions in an optimal manner. That is, the instructions are not necessarily issued in the order specified by the programmer, but in an order that optimally uses the registers, functional units and memory paths of the computer, while at the same time guaranteeing correct results for the computation.
instruction set (n.) the set of low level instructions that a computer is capable of executing. Programs expressed in a high level language must ultimately be reduced to these.
instruction stream (n.) sequence of instructions performed by a computer.
interactive vectorizer (n.) an interactive program to help a user vectorize source code. See also *true ratio*.
interconnection network (n.) the system of logic and conductors that connects the processors in a parallel computer system. Some examples are bus, mesh, hypercube and Omega networks.
interleaved memory (n.) memory divide into a number of modules or banks that can be accessed simultaneously.
internal sort (n.) sorting a table of names contained entirely in primary memory.
interprocessor communication (n.) the passing of data and information among the processors of a parallel computer during the execution of a parallel program.
interprocessor contention (n.) conflicts caused when multiple CPU's compete for shared system resources. For example, memory bank conflicts for a user's code in global memory architectures are caused by other processors running independent applications.
interrupt-driven system (n.) A system whose processes communicate by *message passing* in such a way that when a message is delivered to its destination process it interrupts execution of that process and initiates execution of an interrupt handler process, which stores the message for subsequent retrieval. On completion of the interrupt-handler process (which sets some flag or sends some signal to denote an available message), the original process resumes execution.
interval routing (n.) A *routing* algorithm that assigns an integer identifier to each possible destination and then labels the outgoing *links* of each *node* with a single contiguous interval or window so that a message can be routed simply by sending it out the link in whose interval its destination identifier falls.
invariant (n.) a variable, especially in a DO-loop, that appears only on the right side of an equals sign. The variable is read only, it is never assigned a new value.
invariant expression (n.) an expression, especially in a DO-loop, all of whose operands are *invariant* or constant.
ISO (n.) International Standards Organization, which, among other things, sets standards for programming languages.
isoefficiency(n.) A way to quantify the effect of scaling problem size on an algorithm's *efficiency*. For a given efficiency, the isoefficiency function for an algorithm specifies what size of problem must be solved on a given number of processors to achieve that efficiency.
iterative deepening (n.) use of a D-ply search to prepare for a (D+1)-ply search.

J

Jacobi method (n.) A stationary, iterative method for solving a partial differential equation on a discrete grid. The update of each grid point depends only on the values at neighbouring grid points from the previous iteration. See also *Gauss-Seidel method*.
join(v.) To wait for the termination of one or more descendent processes that were *forked* at some earlier time. See also *spawn*.

K

kernel (n.) A process providing basic services. A service kernel can run on every processor to support minimal *operating system* services, while a *routing* kernel can handle or forward incoming messages.
key (n.) unique object of a search.
knowledge (n.) information plus semantic meaning.
knowledge information processing system(n.) a computer system that processes *knowledge*, rather than data or information.

L

LAN (n.) Local Area Network; any technology for coupling computers together across relatively short (local) distances.
LAPACK (n.) A linear algebra software package, which has been mounted on a wide range of platforms. It evolved from the older *LINPACK* package from *Netlib*. See also *ScaLAPACK*.
latency (n.) The time taken to service a request or deliver a message which is independent of the size or nature of the operation. The latency of a *message passing* system is the minimum time to deliver a message, even one of zero length that does not have to leave the source processor; the latency of a *file system* is the time required to decode and execute a null operation. See also *startup cost*.
lazy evaluation (n.) A Scheduling policy under which no calculation is begun until it is certain that its result is needed. This policy contrasts with the *eager evaluation* used in most programs, but is often used in functional and logic programming. See also *dataflow*, *dependence*, *dependence graph*, *futures*.
light-weight process (n.) A process which executes concurrently with other processes, in the same address space and in an unprotected fashion. Light-weight processes are used by systems such as *MACH* to reduce the overhead of process start-up.
linear speedup (n.) *Speedup* that is directly proportional to the number of processors used. According to *Amdahl's Law*, linear speedup is not possible for a problem that contains any sequential portion, no matter how small. *Gustafson's Law* however, states that linear speedup can be achieved if the problem size is increased as well as the number of processor employed. See also *superlinear speedup*.
linear vector scan (n.) a table lookup algorithm for pipelined vector processors that in a single operation compares a large number of contiguous elements of the table against the key.

link (n.) A one-to-one connection between two processors or *nodes* in a *multicomputer*. See also *bus*.
link loading (n.) The amount of communication traffic carried by a *link*, or by the most heavily loaded link in the system. As link loading increases, both *latency* and *contention* are likely to increase. See also *bisection bandwidth*.
linked array (n.) a data structure designed to allow the joining of various sized lists so that the inserting and deleting of the list elements can be done in parallel without contention.
linked triadic operation (n.) act of executing a pair of vector operations, such as V+S*V as if they were a single, longer operation by taking the output of the first pipelined functional unit and directing it to the second pipelined functional unit, avoiding a trip to main memory and back.
linking (n.) See *chaining*.
LINPACK (n.) A linear algebra software package, which has been mounted on a wide range of platforms. It has now been superceded by *LAPACK*. (n.) also a set of widely quoted *performance benchmarks* based on linear algebra and available from the *National Software Exchange*.
LIPS (n.) logical inferences per second; procedure calls per second in a logic programming system.
live variable(n.) A variable visible to a process and whose value can be changed by actions taken outside that process. For example, if a variable is shared by two processes, one of which can write to it at some point, then that variable is live within the other process. See also *race condition*, *shared variables*.
livelock (n.) A situation in which a process is forever *blocked* because another process has preferential access to a resource needed by both processes.
LIW(n.) Long Instruction Words: the use of long (64 or more bits) instruction words in a processor to improve its ability to pipeline calculations.
load balance(n.) The degree to which work is evenly distributed among available processors. A program executes most quickly when it is perfectly load balanced, that is when every processor has a share of the total amount of work to perform so that all processors complete their assigned tasks at the same time. One measure of load imbalance is the ratio of the difference between the finishing times of the first and last processors to complete their portion of the calculation to the time taken by the last processor.
locality (n.) The degree to which the computations done by a processor depend only on data values held in memory that is close to that processor, or the degree to which computations done on a point in some data structure depend only on values near that point. Locality can be measured by the ratio of local to nonlocal data accesses, or by the distribution of distances of, or times taken by, nonlocal accesses. See also *halo*, *stencil*.
locality of reference (n.) the observation that references to memory tend to cluster. Temporal locality refers to the observation that a particular datum or instruction, once referenced, is often referenced again in the near future. Spatial locality refers to the observation that once a particular location is referenced, a nearby location is often referenced in the near future.

lock (n.) Any device or algorithm whose use guarantees that only one process can perform some action or use some resource at a time.
logarithmic cost criterion (n.) cost criterion that assumes the cost of performing an instruction is proportional to the length of the operands. The integer N requires at least log N + 1 bits of memory, hence the name.
logic (n.) the branch of mathematics that investigates the relationships between premises and conclusions of arguments.
logic program (n.) a set of *Horn clauses*, or procedures.
logical inferences per second (n.) See *LIPS*.
logical time (n.) Elapsed time as seen from within processes. This may differ from *clock time*, because processes can *block* or be suspended during *multitasking* and because they can run at different speeds. The logical times of events only define a partial order on those events.
loop unrolling (n.) A *compiler optimization* technique in which the body of a loop is replicated L times, and the number of iterations of that loop reduced by a corresponding factor L. By lengthening the *basic block* inside the loop, this can increase the scope for *vectorization* and other optimizations.
loose synchronization (adj.) A program running on a *concurrent computer* is said to be running in loose synchronization if the *nodes* are constrained to intermittently synchronize with each other via some communication. Frequently, some global computational parameter such as a time or iteration count provides a natural *synchronization* reference. This parameter divides the running program into compute and communication cycles. See also *synchronous*.
low-order interleaving (n.) a memory interleaving based on the low-order bits of the address.
lower triangular matrix (n.) a matrix with no nonzero elements above the main diagonal.
LU decomposition (n.) a technique where a matrix A is represented as the product of a lower triangular matrix, L, and an upper triangular matrix U. This decomposition can be made unique either by stipulating that the diagonal elements of L be unity, or that the diagonal elements of L and U be correspondingly identical.

M

M-section (n.) a table lookup algorithm for pipelined vector processors that combines features of bisection and linear vector scan.
MACH (n.) an operating system based on Berkely UNIX developed by Carnegie Mellon University.
macropipelining(n.) See *pipelining*.
macrotasking (n.) technique of dividing a computation into two or more large tasks to be executed in parallel. Typically the tasks are subroutine calls executed in parallel.
mailbox (n.) an address used as a source or destination designator in a message.
mapping (n.) often used to indicate an allocation of processes to processors; allocating work to processes is usually called scheduling. See also *load balance*.

marshall(v.) To compact the values of several variables, arrays, or structures into a single contiguous block of memory; copying values out of a block of memory is called unmarshalling. In most *message passing* systems, data must be marshalled to be sent in a single message.
mask (n.) A Boolean array or array-valued expression used to control where a *data parallel* operation has effect; the operation is only executed where array elements are true.
MegaFLOPS (n.) 10^6 *FLOPS*.
memory bank conflict (n.) a condition that occurs when a memory unit receives a request to fetch or store a data item prior to completion of its bank cycle time since its last such request.
memory protection (n.) Any system that prevents one process from accessing a region of memory being used by another. Memory protection is supported both in hardware and by the *operating system* of most serial computers, and by the hardware *kernel* and service kernel of the processors in most parallel computers.
mesh(n.) A *topology* in which nodes form a regular acyclic d-dimensional grid, and each edge is parallel to a grid axis and joins two nodes that are adjacent along that axis. The *architecture* of many *multicomputers* is a two or three dimensional mesh; meshes are also the basis of many scientific calculations, in which each node represents a point in space, and the edges define the neighbours of a node. See also *hypercube*, *torus*.
message passing (n.) A style of interprocess communication in which processes send discrete messages to one another. Some computer *architectures* are called message passing architectures because they support this model in hardware, although message passing has often been used to construct *operating systems* and *network* software for *uniprocessors* and *distributed computers*. See also *routing*.
message typing (n.) The association of information with a message that identifies the nature of its contents. Most *message passing* systems automatically transfer information about a message's sender to its receiver. Many also require the sender to specify a type for the message, and let the receiver select which types of messages it is willing to receive.
message-oriented language (n.) a programming language in which process interaction is strictly through *message passing*.
Metropolis routing(n.) A routing algorithm for *meshes*, in which an ordering is imposed on axes, and messages are sent as far along the most significant axis as they need to go, then as far along the next most significant axis, and so on until they reach their destination. See also *e-cube routing*, *randomized routing*.
microtasking (n.) technique of employing parallelism at the DO-loop level. Different iterations of a loop are executed in parallel on different processors.
middle product method (n.)a method of matrix multiplication in which entire columns of the result are computed concurrently. See also *inner product method*.
MIMD (n.) Multiple Instruction, Multiple Data; a category of *Flynn's taxonomy* in which many instruction streams are concurrently applied to multiple data sets. A MIMD *architecture* is one in which *heterogeneous* processes may execute at different rates.
minimax (n.)algorithm used to determine the value of a game tree.

minimum spanning tree (n.) a spanning tree with the smallest possible weight among all spanning trees for a given graph.
MIPS(n.) one Million Instructions Per Second. A performance rating usually referring to integer or non-floating point instructions. See also *MOPS*.
MISD (n.) Multiple Instruction, Single Data. A member of Flynn's taxonomy almost never used. This category has an ambiguous meaning. It refers to a computer which applies several instructions to each datum. The closest real implementation to this category is a vector computer with an instruction pipeline.
module (n.) a memory bank, often used in the context of interleaved memory.
monitor (n.) a structure consisting of variables representing the state of some resource, procedures to implement operations on that resource, and initialization code.
Monte Carlo (adj.) Making use of randomness. A simulation in which many independent trials are run independently to gather statistics is a Monte Carlo simulation. A search algorithm that uses randomness to try to speed up convergence is a Monte Carlo algorithm.
MOPS(n.) one Million Operations Per Second. Usually used for a general operation, either integer, floating point or otherwise. See also *MIPS*, *FLOPS*.
MOS (n.) Metal oxide on Silicon: a basic technology for fabricating semiconductors. See also *CMOS*, *BiCMOS*.
motherboard (n.) A printed circuit board or card on which other boards or cards can be mounted. Motherboards will generally have a number of slots for other boards, by which means the computer system may be expanded. When all the slots are used up however, it is usually difficult to expand further, and this is the manufacturer's way of telling you to buy a bigger system.
multicast(n.) To send a message to many, but not necessarily all possible recipient processes. See also *broadcast*, *process group*.
multicomputer(n.) A computer in which processors can execute separate instruction streams, have their own private memories and cannot directly access one another's memories. Most multicomputers are *disjoint memory* machines, constructed by joining *nodes* (each containing a microprocessor and some memory) via *links*. See also *architecture*, *distributed computer*, *multiprocessor*, *processor array*.
multigrid method (n.) A method for solving partial differential equations in which an approximate solution on a coarse resolution grid is used to obtain an improved solution on a finer resolution grid. The method reduces long wavelength components of the error or residual by iterating between a hierarchy of coarse and fine resolution grids.
multiprocessor(n.) A computer in which processors can execute separate instruction streams, but have access to a single *address space*. Most multiprocessors are *shared memory* machines, constructed by connecting several processors to one or more memory banks through a *bus* or *switch*. See also *architecture*, *distributed computer*, *multicomputer*, *processor array*.
multiprogramming (n.) the ability of a computer system to time share its (at least one) CPU with more than one program at once. See also *multitasking*.
multitasking(n.) Executing many processes on a single processor. This is usually done by time-slicing the execution of individual processes and performing a context switch each

time a process is *swapped* in or out, but is supported by special-purpose hardware in some computers. Most *operating systems* support multitasking, but it can be costly if the need to switch large *caches* or execution *pipelines* makes *context switching* expensive in time.
mutual exclusion(n.) A situation in which at most one process can be engaged in a specified activity at any time. *Semaphores* are often used to implement this. See also *contention*, *deadlock*, *critical sections*.

N

n-1/2 (n.) (pronounced as n-one-half, usually written with a subscript.) The minimum vector length on which a *pipelinedarchitecture* delivers one half of its theoretical peak performance. The larger n-1/2 is, the longer calculations must be to amortize the *startup cost* of the pipeline. This measure was coined by Hockney and Jesshope in 1988. See also *r-inf*.
NC (n.) The class of parallel algorithms that can run in polylogarithmic (polynomial in the logarithm of the problem size) time on a polynomial number of processors in the *PRAM* model.
necklace (n.) the nodes a data item travels through in response to a sequence of shuffles.
need-predictable (adj.) Used to describe a concurrent algorithm in which the need for but not the nature of point-to-point communication between *nodes* is known prior to program execution. Need predictable problems are *loosely synchronous*.
network(n.) A physical communication medium. A network may consist of one or more *buses*, a *switch*, or the *links* joining processors in a *multicomputer*.
neural network (n.) Man made devices using interconnects and processing capabilities suggested by models of the cortex are termed neural networks. These systems are widely used for optimization problems including content addressable memories and pattern recognition.
NFS (n.) Network File System; a system originally developed by SUN but now widely used, for allowing one machine to access files stored on the file system of another.
node(n.) Basic compute building block of a *multicomputer*. Typically a processor with a memory system and a mechanism for communicating with other processors in the system.
non-blocking(adj.) An operation that does not *block* the execution of the process using it. Usually applied to communications operations, where it implies that the communicating process may perform other operations before the communication has completed. See also *blocking*.
non-deterministic model (n.) a task model in which the execution time of each task is represented by a random variable.
non-uniform memory access (adj.) See *NUMA*.
NP complete (adj.) A term used to describe a problem if it can only be solved in polynomial time by non-deterministic methods. In practice such problems are "hard" and are only solved by a number of heuristic methods that only give approximate answers near to the optimum solution.
NUMA (adj.) Nonuniform memory access; not supporting constant-time read and write operations. In most NUMA *architectures*, memory is organised hierarchically, so that some

portions can be read and written more quickly by some processors than by others. See also *UMA*

O

oblivious(adj.) Working in the same fashion regardless of surroundings or history. An oblivious scheduling strategy always schedules processes in the same way, no matter how much work they have done in the past; an oblivious *routing* algorithm always routes messages in the same direction, regardless of local load. See also *adaptive*.

OEM (n.) Original Equipment Manufacturer; a company which adds components to someone else's computers and sells the result as a complete product.

OLTP (n.) On-line transaction processing; handling transactions (such as deposits and withdrawals) as they occur. An application area of great importance to banks and insurance companies.

Omega network (n.) a composition of shuffle-exchange networks with programmable switches.

operating system (n.) That software responsible for providing standard services and supporting standard operations such as *multitasking* and file access. See also *kernel*.

operation oriented language (n.) a programming language using remote procedure calls as the principle means for interprocess communication and synchronization.

optimal (adj.) Cannot be bettered. An optimal *mapping* is one that yields the best possible *load balance*; an optimal parallel algorithm is one that has the lowest possible time-processor product.

optimization block (n.) a block of code (rarely a whole subprogram, but often a single DO-loop) in which a compiler optimizes the generated code. A few compilers attempt to optimize across such blocks; many just work on each block separately.

optimization problem (n.) a problem whose solution involves satisfying a set of constraints and minimising (or maximising) and objective function.

OR-parallelism (n.) A form of parallelism exploited by some implementations of parallel logic programming languages, in which the terms in disjunctions are evaluated simultaneously, and the parent computation allowed to proceed as soon as any of its children have completed. See also *AND-parallelism*.

OS (n.) See *operating system*.

OSF (n.) Open Software Foundation; an organization established by a number of the major computer manufacturers to set software standards.

output dependence (n.) See *dependence*.

P

packet switching(n.) A routing technique in which intermediate *nodes* wait until they have received the whole of a message before forwarding any of it. Packet switching often requires a large amount of *buffer* space, and *contention* for access to this space can lead to *deadlock*. See also *virtual cut-through*, *wormhole routing*.

page (n.) The smallest unit of a virtual memory system. The system maintains separate virtual-to-physical translation information for each page.
parallel balance point (n.) The point at which using more processors to do a calculation increases the time taken to complete that calculation. See also *Amdahl's Law*, *efficiency*, *Gustafson's Law*, *isoefficiency*, *speedup*.
parallel computation thesis (n.) time on an unbounded parallel model of computation is (polynomially) equivalent to space on a sequential model of computation. (Unproved)
parallel computer (n.) A computer system made up of many identifiable processing units working together in parallel. The term is often used synonymously with *concurrent computer* to include both *multiprocessor* and *multicomputer*. The term concurrent generally dominates in usage in the USA, whereas the term parallel is the more widely used in Europe.
parallel prefix (n.) An operation applying an associative binary operator o to an n-vector V to produce: <(V0)(V0oV1)(V0oV1oV2)...(V0oV1o...oVn)>. Variations of this operation, which is also sometimes called a *scan*, may leave the operations identity element in the first element of the vector, apply the operation downward from the last element of the vector, and so on. See also *reduction*, *scan-vector model*, *segmented parallel prefix operation*.
parallel random access machine (n.) See *PRAM*
parallel slackness (n.) Hiding the *latency* of communication by giving each processor many different task, and having them work on the tasks that are ready while other tasks are *blocked* (waiting on communication or other operations).
parallel unification(n.) finding the set of possible unifications for a goal in parallel.
parallelization (n.) Turning a serial computation into a parallel one. Also sometimes turning a vector computation into a parallel one. This may be done automatically by a parallelising compiler or (more usually) by rewriting (parts of) the program so that it uses some parallel paradigm. See also *dataflow*, *data parallelism*, *futures*, *generative communication*, *message passing*, *shared variables*.
parsing (n.) process whereby a compiler analyzes the syntax of a program to establish the relationship among operators, operands, and other tokens of a program. Parsing does not involve any semantic analysis.
partial cascade sum (n.)parallel algorithms to compute partial sums in logarithmic time.
partial sum (n.) the result from a summation of some subsequence of the elements in a sequence of operands.
partitioning (n.) process of restructuring a program or algorithm into independent computational segments. The goal is to have multiple processors simultaneously work on these independent computational segments.
PDE (n.) partial differential equation.
percentage parallelization (n.) the percentage of processor expenditure processed in parallel on a single job. It is not usually possible to achieve 100 percent of an application's processing time to be shared equally on all processors. See *Amdahl's Law*.
percentage vectorization (n.) The percentage of an application executing in vector mode. This percentage may be calculated as a percentage of CPU time or as the percentage of lines of code (usually Fortran) in vector instructions. The two approaches are

not consistent and may give very different ratings. The first calculation method leads to performance improvement as measured by CPU time, while the second method measures the success rate of the compiler in converting scalar code to vector code. The former is the more meaningful hardware performance measure. See also *vectorize*.

performance model (n.) A formula that predicts the speed, *efficiency*, memory requirements, or other execution characteristics of a program on a particular *architecture*

pipe (n.) A communication primitive which involves the transmission of information through a linearly connected subset of the *nodes* of a *parallel computer*.

pipelining (n.) Overlapping the execution of two or more operations. Pipelining is used within processors by prefetching instructions on the assumption that no branches are going to preempt their execution; in *vector processors*, in which application of a single operation to the elements of a vector or vectors may be pipelined to decrease the time needed to complete the aggregate operation; and in *multiprocessors* and *multicomputers*, in which a process may send a request for values before it reaches the computation that requires them. See also *architecture*.

pivot (n.) A particular element of a matrix during some algorithm. Many matrix algorithms need carefully chosen pivots to improve their numerical accuracy and stability. Pivoting involves arranging that the pivotal element has an appropriate numerical value by reordering and/or swapping rows and columns in the matrix.

polling (adj.) of a communication system. Polling involves a *node* inspecting the communication hardware - typically a flag bit - to see if information has arrived or departed. Polling is an alternative to an *interrupt driven system*. The natural *synchronization* of the nodes imposed by polling is used in the implementation of *blocking* communications primitives.

ports (n.) a variant of mailboxes allowing multiple client processes but only a single server process.

POSIX (n.) A standard of definition of the interface to the UNIX operating system.

PRAM (n.) Parallel random access machine; a theoretical model of parallel computation in which an arbitrary but finite number of processors can access any value in an arbitrarily large *shared memory* in a single time step. Processors may execute different instruction streams, but work *synchronously*. The three most important variations of the PRAM are:

- **EREW** Exclusive read, exclusive write; any memory location may only be accessed once in any one step.

- **CREW** Concurrent read, exclusive write; any memory location may be read any number of times during a single step, but only written to once, with the write taking place after the reads.

- **CRCW** Concurrent read, concurrent write; any memory location may be written to or read from any number of times during a single step. A CRCW PRAM model must define some rule for resolving multiple writes, such as giving priority to the lowest-numbered processor or choosing amongst processors randomly.

The PRAM is popular because it is theoretically tractable and because it gives algorithm

designers a common target. However, PRAMs cannot be emulated *optimally* on all *architectures*. See also *NC*.

preconditioning (n.) a technique for improving the convergence of iterative matrix inversion algorithms such as the *conjugate gradient method*, by transforming the matrix of equation coefficients so that the eigenvalues are redistributed.

private memory (n.) Memory that appears to the user to be divided between many *address spaces*, each of which can be accessed by only one process. Most *operating systems* rely on some *memory protection* mechanism to prevent one process from accessing the private memory of another; in *disjoint memory* machines, the problem is usually finding a way to emulate *shared memory* using a set of private memories. See also *virtual shared memory*.

procedure oriented language (n.) a programming language in which process communication and synchronization are accomplished through the use of of shared variables.

process (n.) the fundamental entity of the software implementation on a computer system. A process is a sequentially executing piece of code that runs on one processing unit of the system.

process creation (n.) The act of *forking* or *spawning* a new process. If a system permits only static process creation, then all processes are created at the same *logical time*, and no process may interact with any other until all have been created. If a system permits dynamic process creation, then one process can create another at any time. Most first and second generation *multicomputers* only supported static process creation, while most *multiprocessors*, and most *operating systems* on *uniprocessors*, support dynamic process creation. See also *configuration*, *generative communication*.

process flow graph (n.) an *acyclic* directed graph in which vertices represent processes and edges represent execution constraints.

process group (n.) A set of processes that can be treated as a single entity for some purposes, such as *synchronization* and *broadcast* or *multicast* operations. In some parallel programming systems there is only one process group, which implicitly contains all processes; in others, programmers can assign processes to groups statically when configuring their program, or dynamically by having processes create, join and leave groups during execution.

process migration (n.) Changing the processor responsible for executing a process during the lifetime of that process. Process migration is sometimes used to dynamically *load balance* a program or system.

processor array (n.) A computer that consists of a regular *mesh* of simple processing elements, under the direction of a single control processor. Processor arrays are usually *SIMD* machines, and are primarily used to support *data parallel* computations. See also *array processor*, *vector processor*.

processor overloading (n.) *Mapping* many processes to a single processor, so that *parallel slackness* can be exploited.

Prolog (n.) a language for logic programming.

pseudovector (n.) a scalar temporary variable.

Q

QCD (n.) Quantum Chromodynamics; a model of the behaviour of matter on sub-nuclear scales, the simulation of which is very hungry of computing power.

R

r-inf (n.) (pronounced r-inf, often written with a subscript) The performance a *pipelinedarchitecture* would deliver on an infinitely long vector; that is, the performance of such an architecture when *startup costs* are not considered. This parameter was coined by Hockney and Jesshope in 1988. See also *n-1/2*.

race condition(n.) A situation in which the final result of operations being executed by two or more processes depends on the order in which those processes execute. For example, if two processes A and B are to write different values VA and VB to the same variable, then the final value of the variable is determined by the order in which A and B are *scheduled*.

RAID (n.) Redundant array of inexpensive disks; a *file system* containing many disks, some of which are used to hold redundant copies of data or error correction codes to increase reliability. RAIDS are often used as parallel access file systems, where the sheer size of storage capacity required precludes using more conventional (but more expensive) disk technology.

RAM (n.) Random Access Memory; computer memory which can be written to and read from in any order. See also *DRAM*, *SRAM*.

random uniform game tree (n.) a game tree whose terminal node values are randomly chosen from some uniform distribution.

randomized routing (n.) A *routing* technique in which each message is sent to a randomly chosen *node*, which then then forwards it to its final destination. Theory and practice show that this can greatly reduce the amount of *contention* for access to *links* in a *multicomputer*.

rank (n.) row of a *butterfly network*.

RDMS (n.) Relational Database Management System; software to manage a database in which data are stored by attribute.

ready list (n.) *OS* list containing ready-to-run processes.

reasonable (adj.) a parallel model is said to be reasonable if the number of processors each processor can communicate with directly is bounded by a constant.

recurrence (n.) a relationship in a DO-loop whereby a computation in one iteration of the loop depends upon completion of a previous iteration of the loop. Such dependencies inhibit vectorization.

reduced instruction set computer (adj.) See *RISC*

reduction operation (n.) An operation applying an associative and commutative binary operator to a list of values, See also *parallel prefix*.

redundant array of inexpensive disks (n.) See *RAID*

redundant computation (n.) Calculations that are carried out more than once or by more than one processor. Computations may be done redundantly because it is cheaper to have every processor calculate a value for itself than to have one processor calculate the

value and then *broadcast* it, or because processes may not have enough memory to store all the values they calculate and may need to overwrite some during execution.
refute (v.) to make a move that causes an *alpha-beta* cutoff.
relaxation method (n.) A type of *indirect method* in which the values making up a trial solution to an equation are repeatedly updated according to some rule until convergence criteria are met. See also *direct method*
remote procedure call (n.) a structured implementation of a client-server interaction.
rendezvous (n.) when the server side of a *remote procedure call* is specified by using an accept statement or similar construct.
reply message (n.) passing of results back to the client in a *remote procedure call.*
RGB (adj.) Red-Green-Blue; the most common form of colour display hardware.
ring (n.) A *topology* in which each *node* is connected to two others to form a closed loop. See also *chain*, *Hamiltonian.*
RISC (adj.) Reduced instruction set computer; a computer that provides only a few simple instructions but executes them extremely quickly. RISC machines typically rely on instruction prefetching and *caching* to achieve higher performance than *CISC* machines. The term is also applied to software designs that give users a small number of simple but efficient operations.
ROM (n.) Read Only Memory; a computer memory which cannot be written to during normal operation.
routing (n.) The act of moving a message from its source to its destination. A routing technique is a way of handling the message as it passes through individual nodes. See also *e-cube routing*, *interval routing*, *Metropolis routing*, *packet switching*, *randomized routing*, *virtual cut-through*, *wormhole routing.*
routing algorithm (n.) a rule for deciding, at any intermediate *node*, where to send a message next. See also *routing.*
routing kernel (n.) See *kernel.*
rule (n.) in the context of logic programming, a rule is a *Horn clause* with a head and a body.

S

satisfiable (adj.) true under the rules of logic.
SAXPY (n.) elemental *BLAS* operation involving "constant (a) times vector (x) plus vector (y)". The S refers to Fortran Single precision. SAXPY and related BLAS operations are often implemented by the hardware manufacturer as part of the system software and the execution time for such operations has been used a a primitive benchmark of a high performance computing system.
scalable (adj.) Capable of being increased in size, or more accurately, capable of delivering an increase in performance proportional to an increase in size. A scalable *architecture* is one that can be used as a design for arbitrarily large machines, or one whose increase in performance is linear in the amount of hardware invested. The term is also applied to programming systems, although its meaning is less clear in these cases. It is generally used to imply that the methods and algorithms employed in such systems are in principle

capable of performing correctly equally well on large and small hardware systems. See also *Gustafson's Law*

ScaLAPACK (n.) A linear algebra software package, which has been mounted on a wide range of platforms. This is a version of *LAPACK* suitable for *distributed memory* computer systems. The *software* is available from the *National Software Exchange*. See also *LAPACK*.

scalar processor (n.) A computer in the traditional Von Neumann sense of operating only on scalar data. See also *uniprocessor*, *vector processor*, *array processor*.

scalar temporary (n.) a compiler created scalar variable that holds the value of a vectorizable expression on each iteration of a DO-loop.

scan (n.) See *parallel prefix*.

scan-vector model (n.) A theoretical model of parallel computing in which a *scalar processor* and a *vector processor* have access, respectively to a memory holding scalar values and a memory holding vectors of arbitrary length. Vector operations take either a single time step or a time proportional to the logarithm of the number of elements. See *parallel prefix*, *data parallelism*, *reduction operation*.

scattered decomposition (n.) See *decomposition*.

scheduling (n.) Deciding the order in which the calculations in a program are to be executed, and by which processes. Allocating processes to processors is usually called *mapping*. See also *load balance*.

scoreboard (n.) A hardware device that maintains the state of machine resources to enable instructions to execute without conflict at the earliest opportunity.

SCSI (n.) Small Computer Systems Interface; a hardware standard for interfacing to devices such as disks.

secondary memory (n.) a larger but slower memory than primary memory. Access to secondary memory often requires special instructions, such as *I/O* instructions.

segmented parallel prefix operation (n.) A *parallel prefix* operation in which the vector being operated on is divided into segments, and the operator is applied to each segment as if it were a separate vector. This is usually implemented by supplying a separate vector of segment lengths.

self-scheduling (adj.) Automatically allocating work to processes. If T tasks are to be done by P processors, and $P < T$, then they may be self-scheduled by keeping them in a central pool from which each processor claims a new job when it finishes executing its old one. See also *task farming*.

semaphore (n.) A data type for controlling concurrency. A semaphore can be initialised to any non negative integer value. After that, only two operations may be applied to it: "signal" which increments the semaphore's value by one, and "wait" which *blocks* its caller until the semaphore's value is greater than zero, then decrements the semaphore. The value of a semaphore typically represents the amount of some resource that is currently available, while waiting on a semaphore forces processes to block until some of that resource can be claimed. A binary semaphore is one that can only take on the values 0 and 1.

sequential bottleneck (n.) A part of a computation for which there is little or no parallelism. See also *Amdahl's Law*.

sequential computer (n.) synonymous with a *Von Neumann architecture* computer and is a "conventional" computer in which only one processing element works on a problem at a given time. See also *uniprocessor*.
serialize (v.) To put potentially concurrent operations in a strictly sequential order. If concurrent processes must claim a *lock* before doing some operation, for example, then their operations will be serialized.
service kernel (n.) See *kernel*.
set associative (n.) A cache structure in which all tags in a particular set are compared with an access key in order to access an item in cache. The set may have as few as one element or as many elements as there are lines in the full cache.
shared memory (n.) Memory that appears to the user to be contained in a single *address space* and that can be accessed by any process. In a *uniprocessor* or *multiprocessor* there is typically a single memory unit, or several memory units interleaved to give the appearance of a single memory unit. See also *disjoint memory*, *distributed memory*.
shared variables (n.) Variables to which two or more processes have access, or the model of parallel computing in which interprocess communication and *synchronization* are managed through such variables. See also *data parallelism*, *futures*, *generative communication*, *live variable*, *message passing*.
short necklace (n.) a *necklace* of length less than k in a shuffle exchange network containing 2^k nodes.
shortstopping (v.) using the output of a functional unit before it is routed back to memory.
shuffle exchange network (n.) A *topology* containing N = 2^L *nodes*, each of which is labeled by a unique L-bit integer. If two nodes have labels <—IL...I0—> and <—JL...J0—>, then I and J are connected if Ik=Jk for 1<=k<(L-1) and I0<>J0, or if J is a left or right cyclic shift of I. See also *butterfly*, *hypercube*.
SIMD (adj.) Single instruction, multiple data; a category of *Flynn's taxonomy* in which a single instruction stream is concurrently applied to multiple data sets. A SIMD *architecture* is one in which *homogeneous* processes *synchronously* execute the same instructions on their own data, or one in which an operation can be executed on vectors of fixed or varying size. See also *array processor*, *processor array*, *vector processor*.
SIMD-CC (n.) cube-connected (*hypercube*) *SIMDarchitecture*.
SIMD-CCC (n.) cube connected cycles SIMD *architecture*. See also *SIMD-CC*.
SIMD-MC (n.) mesh connected *SIMDarchitecture*. Superscript refers to the number of dimensions in the architecture.
SIMD-P (n.) *SIMDarchitecture* with pyramid processor organization.
SIMD-PS (n.) *SIMDarchitecture* with perfect shuffle connections.
SIMD-SM (n.) shared memory *SIMDarchitecture*. Two processors may not read from or write into the same shared memory location during the same instruction.
SIMD-SM-R (n.) shared memory *SIMDarchitecture*. Two processors may read from the same memory location during an instruction, but concurrent writing into the same location is forbidden.

SIMD-SM-RW (n.) shared memory *SIMDarchitecture*. Concurrent reading from and writing to the same memory location is allowed.
simulated annealing (n.) An optimization technique introduced by Kirkpatrick in 1983 which uses statistical physics methods to find an approximate optimal solution to a problem. Typically a thermodynamic analogy is used for the model system under study and the task of finding an optimal solution is mapped to that of finding the ground state of the thermodynamic system.
single instruction, multiple data (adj.) See *SIMD*.
single instruction, single data (adj.) See *SISD*.
single program, multiple data (adj.) See *SPMD*.
single-source shortest-path problem (n.) problem of finding the shortest path from a single designated vertex (the source) to all the other vertices in a weighted, directed graph.
SISD (adj.) Single instruction, single data; a category of *Flynn's taxonomy* in which a single instruction stream is serially applied to a single data set. Most *uniprocessors* are SISD machines.
software lockout (n.) See *lock*.
SOR (n.) Successive over-relaxation is a technique for accelerating the convergence of *relaxation methods* for solving sets of simultaneous linear equation, Ax=b. It typically involves adding an appropriate multiple of the unit matrix to the matrix of coefficients A.
space complexity (n.) space used up by an algorithm as a function of problem size.
space sharing (n.) Dividing the resources of a parallel computer among many programs so they can run simultaneously without affecting one another's performance. See also *time sharing*.
spanning tree (n.) A *tree* containing a subset of the *links* in a graph which reaches every node in that graph. A spanning tree can always be constructed so that its depth (the greatest distance between its root and any leaf) is no greater than the *diameter* of the graph. Spanning trees are frequently used to implement *broadcast* operations.
SPARC (n.) Scalable Processor ARChitecture; a family of chips which can be manufactured using a variety of technologies, but still be compatible in some ways.
sparse matrix (n.) A matrix with the majority of its elements equal to zero. See also *full matrix*.
spawn (v.) To create a new process with arbitrary initial memory contents and instructions. See also *fork*.
speedup (n.) The ratio of two program execution times, particularly when times are from execution on 1 and P *nodes* of the same computer. Speedup is usually discussed as a function of the number of processors, but is also a function (implicitly) of the problem size. See also *Amdahl's Law*, *Gustafson's Law*, *isoefficiency*, *optimal*.
spin lock (n.) an implementation of the lock primitive that causes a processor to retest a semaphore until it changes value. Busy-waits will spin until the lock is free.
spinning (adj.) a process waiting for the value of a *spin lock* to change is said to be spinning.
SPMD (adj.) Single program, multiple data; a category sometimes added to *Flynn's taxonomy* to describe programs made up of many instances of a single type of process, each

executing the same code independently. SPMD can be viewed either as an extension of *SIMD*, or as a restriction of *MIMD*. See also *process group*, *SISD*.

SQL (n.) Standard Query Language; a standard for adding data to, or recovering data from, databases.

SRAM (n.) Static *RAM*; memory which stores data in such a way that it requires no memory refresh cycle and hence have low power consumption. Generally this type of RAM is faster but more expensive than *DRAM*.

startup cost (n.) The time taken to initiate any transaction with some entity. The startup cost of a *message passing* system, for example, is the time needed to send a message of zero length to nowhere. See also *latency*.

static channel naming (n.) a *message passing scheme* in which source and destination designators are fixed at compile time.

static decomposition (n.) task allocation policy that assumes tasks and their precedence relations are known before execution.

stencil (n.) A pattern of data accesses used when updating the values in a *mesh*. A stencil is usually represented as a grid around a central point, which indicates the location of the value being updated.

stream parallelism (n.) a pipelined variant of AND parallelism.

stream unit (n.) transmits vectors into the vector arithmetic section on some vector CPUs.

strength reduction (n.) a process whereby a compiler attempts to replace instructions with less time-costly instructions that produce identical results. For example in Fortran, X**2 might be automatically replaced by X*X.

stride (n.) a term derived from the concept of walking or striding through data from one location to the next. The term is often used in relation to vector storage. A mathematical vector is represented in a computer as an array of numbers. Most computers use contiguous storage of vectors, locating them by the address of the first word and by the vector length. Many applications in linear algebra however, require load and store of vectors with components that do not reside contiguously in memory, such as the rows of a matrix stored in column order. The row vectors are spaced in memory by a distance or stride of the leading dimension of the array representing the matrix. Some vector computers allow vector fetching and storing to occur with randomly stored vectors. An index vector locates successive components. This is often useful in storing the nonzero elements of a sparse vector.

striped (adj.) A *file system* that distributes files across disks by distributing individual blocks, often at the single-bit level. See also *declustered*.

stripmining (n.) a process used by a compiler on a register-to-register vector processor whereby a DO-loop of long or variable iteration count is executed in strips which are the length of a vector register, except for a remainder strip whose length is less.

strong search (n.) an algorithm that searches for a given key, locks the node associated with that key, and returns the node.

strongly ordered game tree (n.) game tree with the following properties: (1) 70 percent of the time the first move chosen from any nonterminal node is the best move, and (2) 90 percent of the time the best move from any nonterminal node is one of the first 25 percent

of the moves searched.

subcube (n.) term for a subset of *nodes* of a *hypercubehypercube*. The hypercube architecture has a natural recursive definition so that a cube of dimension d1 includes within it 2^(d1-d2) lower dimensional sets of nodes, each of which is itself a hypercube of dimensionality d2. These subsets of nodes are known as subcubes.

subgraph (n.)given a graph, a subgraph is another graph whose vertices and edges are in the original graph.

successive over relaxation (n.) See *SOR*.

supercomputer (n.) a time dependent term which refers to the class of most powerful computer systems worldwide at the time of reference. See also *concurrent computer*, *parallel computer*.

superlinear speedup (n.) *Speedup* that is greater than an amount proportional to the number of processors used. While super-linear speedup is theoretically impossible, in practice it may occur because distributing a problem among many processors may increase the effective total size of the *cache* being used, or because distribution may change the order in which nondeterministic operations are carried out, which can lead to earlier termination of the program.

superword (n.) a term used on some computers to describe a group of eight 64-bit words, or alternatively, sixteen 32-bit half-words. The memory units on such machines, generally fetch and store data in superwords (also called swords), regardless of the size of the data item referenced by the user program.

swap (v.) To exchange two items. The term refers to swapping a section of real memory (contents) for a section of *virtual memory*.

switch (n.) A physical communication medium containing nodes that perform only communications functions. Examples include crossbar switches, in which N+M *buses* cross orthogonally at NM switching points to connect N objects of one type to M objects of another, and multistage switches in which several layers of switching *nodes* connect N objects of one type to N objects of another type. See also *butterfly*, *combining*, *network*, *shuffle exchange network*.

synchronization (n.) The act of bringing two or more processes to known points in their execution at the same *clock time*. Explicit synchronization is not needed in *SIMD* programs (in which every processor either executes the same operation as every other or does nothing), but is often necessary in *SPMD* and *MIMD* programs. The time wasted by processes waiting for other processes to synchronize with them can be a major source of inefficiency in parallel programs. See also *asynchronous*, *barrier synchronization*, *synchronous*.

synchronous (adj.) Occurring at the same *clock time*. For example, if a communication event is synchronous, then there is some moment at which both the sender and the receiver are engaged in the operation. See also *asynchronous*.

systolic (adj.) Driven by the availability of data or other resources. In a systolic system, processes execute operations *synchronously* as their inputs become available.

systolic array (n.) A class of *parallel computers* with a fixed *array* of fine grain *nodes* through which data is pumped by a master or control processor.

T

task farming (n.) A technique for implementing *self-scheduling* calculations. In a task farm, a source process generates a pool of jobs, while a sink process consumes results. In between, one or more worker processes repeatedly claim jobs from the source, turn them into results, dispatch those results to the sink, and claim their next jobs. If the number of jobs is much greater than the number of workers, task farming can be an effective way to *load balance* a computation.

TeraFLOPS (n.) 10^12 *FLOPS*.

thrashing (n.) a phenomenon of virtual memory systems that occurs when the program, by the manner in which it is referencing its data and instructions, regularly causes the next memory locations referenced to be overwritten by recent or current instructions. The result is that referenced items are rarely in the machines physical memory and almost always must be fetched from secondary storage, usually a disk. Cache thrashing involves a similar situation between cache and physical memory.

thread (n.) a lightweight or small granularity process.

throughput(n.) number of results produced per unit time.

tiling (n.) A regular division of a *mesh* into patches, or tiles. Tiling is the most common way to do *geometric decomposition*.

time sharing (adj.) Dividing the effort of a processor among many programs so they can run concurrently. Time sharing is usually managed by an *operating system*. See also *space sharing*.

time-processor product (n.) The product of the time taken to execute a program and the number of processors used to achieve that time, often used as a measure of goodness for parallel algorithms. See also *Amdahl's Law*, *efficiency*, *Gustafson's Law*, *speedup*.

TLB (n.) translation look-aside buffer; the memory cache of the most recently used page table entries within the memory management unit.

topology (n.) A family of graphs created using the same general rule or that share certain properties. The processors in a *multicomputer*, and the circuits in a *switch*, are usually laid out using one of several topologies, including the *mesh*, the *hypercube*, the *butterfly*, the *torus* and the *shuffle exchange network*. See also *bisection bandwidth*, *diameter*.

torus (n.) A *topology* in which *nodes* form a regular cyclic d-dimensional grid, and each edge is parallel to a grid axis and joins two nodes that are adjacent along that axis. The *architecture* of some *multicomputers* is a two or three dimensional torus. See also *hypercube*, *mesh*.

trace scheduling(n.) A *compiler optimization* technique that *vectorizes* the most likely path through a program as if it were a single *basic block*, includes extra instructions at each branch to undo any ill effects of having made a wrong guess, vectorizes the next most likely branches and so on.

transposition table (n.) a hash table that stores previously evaluated game positions.

transputer (n.) A single integrated circuit which contains a *CPU*, communications *links*, and some *cache* memory. The name transputer refers to a proprietary series of chips manufactured by Inmos, although other *node* chips have had similar characteristics.

tree (n.) a connected, undirected, acyclic graph. The most commonly encountered tree in computer science is the regular binary tree, in which a root *node* has two children but no parent, each interior node has a single parent and two children, and leaf nodes have a single parent but no children.
true dependence (n.) See *dependence*.
true ratio (n.) the frequency with which the "true" branch of a Fortran IF-test occurs. If the true ratio is know at compile time, some compilers can take advantage of this knowledge. However the true ratio is often data dependent and cannot effectively be dealt with automatically. See also *interactive vectorizer*.
tuple (n.) An ordered sequence of fixed length of values of arbitrary types. Tuples are used for both data storage and interprocess communication in the *generative communication* paradigm. See also *tuple space*.
tuple space (n.) A repository for tuples in a *generative communication* system. Tuple space is an *associative memory*.

U

UART (n.) Universal Asynchronous Receive-Transmit; a standard protocol for device drivers.
UMA (adj.) Uniform memory access; permitting any memory element to be read or written in the same, constant time. See also *NUMA*.
undirected graph (n.)a graph whose edges have no orientation.
unification (v.) instantiation of a variable with a value.
uniform cost criterion (n.) the assumption that every instruction takes one unit of time and every register requires one unit of space.
uniform memory access (adj.) See *UMA*.
uniprocessor (n.) A computer containing a single processor. The term is generally synonymous with *scalar processor*.
UNIX (n.) an *OS* originally developed by AT&T which, in various incarnations, is now available on most types of supercomputer.
unneeded store (n.) situation resulting when two or more stores into the same memory location without intermediate reads occur in an optimization block, especially within a DO-loop, such that only the last store need actually be performed.
utilization (n.) percentage of time a processor spends executing useful tasks during execution of a program.

V

valence (n.) The number of edges connected to a vertex in a graph; for example, every *node* in a regular square *mesh* has a valence of4. Confusingly, valence also means the number of branches below a *tree* node, which is one fewer than the number of edges incident to that node - every node in a binary tree has a valence of 2. The term *arity* is sometimes also used in this sense.
vector (n.) an ordered list of items in a computer's memory. A simple vector is defined as

having a starting address, a length, and a stride. An indirect address vector is defined as having a relative base address and a vector of values to be applied as indices to the base.
vector processor (n.) A computer designed to apply arithmetic operations to long vectors or arrays. Most vector processors rely heavily on *pipelining* to achieve high performance. See also *array processor*.
vector register (n.) a storage device that acts as an intermediate memory between a computer's functional units and main memory.
vectorize (v.) To transform a sequence of identical arithmetic operations into a single instruction. See also *array processor*, *vector processor*.
vertex (n.) component of a graph, also sometimes called a node.
vertical processing (n.) processing a two dimensional array column by column.
virtual channel (n.) A logical point-to-point connection between two processes. Many virtual channels may *time share* a single *link* to hide *latency* and to avoid *deadlock*. See also *wormhole routing*.
virtual concurrent computer (n.) A computer system that is programmed as a *concurrent computer* of some number of *nodes* P, but which is implemented on either a real concurrent computer of some number of nodes less than P or on a *uniprocessor* running software to emulate the environment of a concurrent machine. Such an emulation system is said to provide virtual nodes to the user.
virtual cut-through (n.) A technique for *routing* messages in which the head and tail of the message both proceed as rapidly as they can. If the head is *blocked* because a *link* it wants to cross is being used by some other message, the tail continues to advance, and the message's contents are put into *buffers* on intermediate *nodes*. See also *packet switching*, *wormhole routing*.
virtual memory (n.) A system that stores portions of an *address space* that are not being actively used in in some medium other than main high-speed memory, such as a disk or slower auxiliary memory medium. When a reference is made to a value not presently in main memory, the virtual memory manager must *swap* some values in main memory for the values required. Virtual memory is used by almost all *uniprocessors* and *multiprocessors*, but is not available on some *array processors* and *multicomputers*, which still employ real memory storage only on each *node*.
virtual shared memory (n.) Memory that appears to users to constitute a single *address space*, but which is actually physically disjoint. Virtual shared memory is often implemented using some combination of hashing and local *caching*.
VLIW (n.) Very Long Instruction Word; the use of extremely long instructions (256 bits or more) in a computer to improve its ability to chain operations together.
VLSI (adj.) Very Large Scale Integration; applied to technologies capable of putting hundreds of thousands or more components on a single chip, or sometimes applied to those chips so manufactured.
VMS (n.) Virtual Machine System; an operating system developed by DEC and widely used on VAX machines. Popularity of this OS is probably waning in favour of UNIX like systems.
Von Neumann architecture (n.) Used to describe any computer which does not employ

concurrency or parallelism. Named after John Von Neumann (1903-1957) who is credited with the invention of the basic architecture of current sequential computers.

W

WARPS (n.) Words accessed randomly per second; a measure of memory access performance, equal to the rate of uniformly random accesses across the whole of the *address space* visible to a process that a machine supports. See also *FLOPS*.

weak search (n.) a search algorithm that searches for a key and returns the node that contained the key at the time it was examined. Weak searches are not guaranteed to provide an up-to-date result. See also *strong search*.

weight (n.) a real number assigned to an edge in a *weighted graph*.

weight matrix (n.) a matrix indicating, for each pair of vertices I and J, the weight of the edge from vertex I to vertex J.

weighted graph (n.) a graph with a real number assigned to each edge.

working set (n.) Those values from *shared memory* that a process has copied into its *private memory*, or those pages of *virtual memory* being used by a process. Changes a process makes to the values in its working set are not automatically seen by other processes.

wormhole routing (n.) A technique for *routing* messages in which the head of the message establishes a path, which is reserved for the message until the tail has passed through it. Unlike *virtual cut-through*, the tail proceeds at a rate dictated by the progress of the head, which reduces the demand for intermediate *buffering*. See also *packet switching*.

worst-case space complexity (n.) greatest amount of space used by an algorithm, over all possible inputs of a given size.

wrap-around scalar (n.) a scalar variable whose value is set in one iteration of a DO-loop and referenced in a subsequent iteration and is consequently recursive. All common reduction-function scalars are wrap-around scalars and usually do not prevent vectorization. All other wrap-around scalars usually do prevent vectorization of the loop in which they appear.

write-in cache (n.) a cache in which writes to memory are stored in the cache and written to memory only when a rewritten item is removed from cache. This is also referred to as write-back cache.

write-through cache (n.) a cache in which writes to memory are performed concurrently both in cache and in main memory.

X

Y

Z

Ken Hawick, hawick@npac.syr.edu