# Glossary

Underlined terms within entries are also defined in the glossary.

**2-Opt:** Simple iterative improvement algorithm for the Travelling Salesman Problem that is based on the 2-exchange neighbourhood on edges of the given graph. The candidate tours obtained by 2-opt local search are called 2-*optimal* (or 2-*opt*) tours.

**3-Opt:** Well-known iterative improvement algorithm for the Travelling Salesman Problem (TSP) that is based on the 3-exchange neighbourhood on edges of the given graph. 3-opt local search provides the basis for a number of high-performance stochastic local search algorithms for the TSP. The candidate tours obtained by 3-opt local search are called 3-*optimal* (or 3-*opt*) tours.

***k*-Exchange Neighbourhood:** Neighbourhood in which a candidate solution differs from any of its direct neighbours in up to $k$ solution components. Prominent examples include the 2- and 3-*exchange neighbourhoods* for the Travelling Salesman Problem; 1-*exchange neighbourhoods* are widely used in stochastic local search algorithms for many combinatorial problems, including the Satisfiability Problem and the finite discrete Constraint Satisfaction Problem.

***k*-Flip Neighbourhood:** $k$-exchange neighbourhood for the Satisfiability Problem (SAT) or the Maximum Satisfiability Problem (MAX-SAT), under which the direct neighbours of a candidate solution are precisely those truth assignments that are obtained by flipping (i.e., changing) the value of up to $k$ propositional variables. Most stochastic local search algorithms for SAT and MAX-SAT are based on the 1-*flip neighbourhood*.

**Active Schedule:** A schedule in which no operation can be completed earlier by any change in the processing sequence of any machine without delaying other operations.

**Adaptive Iterative Construction Search (AICS):** Stochastic local search method that is based on multiple iterations of a constructive search procedure; the construction is guided by heuristic information and by experience gained from past iterations. The latter is represented in the form of *weights* associated with the elementrary decisions that can be made during the construction process; these weights are adapted during the search. AICS is conceptually closely related to Ant Colony

Optimisation (ACO), and the pheromone trails in ACO correspond to the weights in AICS.

**Algorithm Portfolio:** Collection of algorithms that are simultaneously or selectively applied to a given problem instance. Compared to their constituting algorithms, portfolios can often solve combinatorial problems more robustly, particularly in situations where different algorithms are likely to perform best on various types of problem instances, depending on partially unknown features of the instance under consideration. Because of the additional uncertainty in run-time, portfolios of Las Vegas algorithms are particularly interesting. Portfolios of stochastic local search algorithms can be adequately represented using co-operative GLSM models.

**Annealing Schedule:** Often also called *cooling schedule*; a mapping used in Simulated Annealing (SA) that determines for each run-time $t$ the value of the temperature parameter, $T(t)$. Annealing schedules are commonly specified by an initial temperature, $T(0)$, a temperature update scheme, a number of iterations to be performed at each temperature and a termination condition. Although the name suggests a monotonic decrease of temperature over time, annealing schedules can be arbitrary functions and may contain segments of increasing or constant temperature.

**Ant Colony Optimisation (ACO):** Stochastic local search method that is inspired by the pheromone trail following behaviour of some ant species. In analogy to the biological example, ACO is based on the indirect communication of a colony of simple agents, called (artificial) *ants*, mediated by (artificial) *pheromone trails*. The pheromone trails in ACO serve as a distributed, numerical memory that the ants use to probabilistically construct candidate solutions to the given problem instance; like the weights in Adaptive Iterated Construction Search, the pheromone trails are adapted during the search process and represent the collective experience of the ants. A large number of different ACO algorithms has been proposed, including $\mathcal{MAX}$-$\mathcal{MIN}$ Ant System and Ant Colony System. Most of the best-performing ACO algorithms for combinatorial problems use subsidiary local search to improve the candidate solutions constructed by the ants.

**Ant Colony System (ACS):** Ant Colony Optimisation algorithm that uses an aggressive construction procedure in which, with high probability, deterministic choices are made instead of the usual stochastically biased choice. Furthermore, unlike many other ACO algorithms, ACS performs additional updates of the pheromone trails during the construction of candidate solutions.

**Approximate (Optimisation) Algorithm:** Incomplete search algorithm for an optimisation problem, that is, an algorithm for solving an optimisation problem that is not guaranteed to find an optimal solution, even if run for an arbitrarily long (finite) amount of time.

**Approximation Algorithm:** Algorithm for solving an optimisation problem that has provable performance guarantees, typically in the form of constant bounds on the worst-case approximation ratio. (The term is sometimes used incorrectly to refer to optimisation algorithms that may return suboptimal candidate solutions or solutions but do not have performance guarantees.)

**Approximation Ratio:** Performance criterion for optimisation algorithms; for an algorithm $A$, the approximation ratio $r$ on a given instance $\pi'$ of an optimisation problem $\Pi'$ with objective function $f$ is defined as $r := \max\{\hat{q}/q^*,\ q^*/\hat{q}\}$, where $\hat{q}$ is the best solution quality achieved by $A$ on $\pi'$, and $q^*$ is the optimal solution quality of $\pi'$. The (worst-case) approximation ratio of $A$ on problem $\Pi'$, that is, the maximum approximation ratio of $A$ over all instances $\pi' \in \Pi'$, plays an important role in computational complexity theory.

**Arc Consistency:** Property of an instance of the Constraint Satisfaction Problem (CSP); a CSP instance is arc consistent if, and only if, there is no variable $x$ whose domain includes a value $d$ such that $x$ occurs in a constraint $C$ with no satisfying tuple for which $x$ has value $d$. Arc consistency can always be enforced in polynomial time w.r.t. to the number and maximal domain size of the variables in a given CSP instance. Procedures for enforcing arc consistency play an important role in the context of systematic search algorithms and stochastic local search algorithms for the CSP.

**Aspiration Criterion:** Condition used in a tabu search algorithm to override the tabu status of candidate solutions or solution components, for example, when the respective search step leads to an improvement in the current incumbent candidate solution.

**Autocorrelation Coefficient:** Measure of ruggedness of a search landscape; empirically determined from the *autocorrelation function* of an uninformed random walk. Intuitively, larger autocorrelation coefficients indicate smoother landscapes. Closely related to the autocorrelation coefficient is the correlation length, which has a similar intuitive interpretation.

**Backtracking:** Algorithmic technique in which a search algorithm, upon encountering a 'dead end' from which further search is impossible or unpromising, reverts to an earlier point in its search trajectory. Many systematic search algorithms are based on combinations of constructive search methods with backtracking.

**Basin Partition Tree (BPT):** Represents the basin structure of a search landscape; a BPT $T$ forms a complete partition of a given landscape $S$, that is, the vertices of $T$ represent disjoint subsets of $S$, and every position in $S$ is represented by exactly one vertex in $T$. BPTs can be seen as abstractions of plateau connection graphs and are closely related to basin trees.

**Basin Tree (BT):** Represents the basin structure of a search landscape; the vertices of a BT represent the *barrier-level basins* of a given landscape, and the edges indicate immediate containment between the respective basins. BTs are closely related to basin partition trees.

**Benchmark Instance:** Problem instance used for the empirical evaluation and characterisation of algorithmic behaviour. Sets of benchmark instances (also referred to as *benchmark suites*) are often made available via on-line benchmark libraries.

**Biergarten:** Open air pub where people enjoy a refreshing *Maß of Weißbier* (a litre of a specific type of Bavarian beer), (light) supper and good company, typically whilst sitting under large, old chestnut trees on a mild summer evening; Biergärten are an essential part of Southern German culture. Many of the ideas and concepts described in this book were at some point discussed between its authors during their many pleasant visits to a local Biergarten.

**Bivariate Run-Time Distribution:** Joint probability distribution of run-time and solution quality that characterises the behaviour of an optimisation Las Vegas algorithm.

**Branch & Bound:** Systematic search method for combinatorial optimisation problems that exploits *upper and lower bounding techniques* for effectively pruning the search tree of a given problem instance. In particular, the search can be pruned at any partial solution whose lower bound exceeds the current upper bound on the quality of an optimal solution for the given problem instance (in the case of a minimisation problem). In the context of branch & bound algorithms for minimisation problems, stochastic local search algorithms can be used for obtaining upper bounds on the optimal solution quality of a given problem instance. Branch & bound algorithms are amongst the best-performing systematic search methods for many hard combinatorial optimisation problems, such as the Maximum Satisfiability Problem.

**Branch & Cut:** Method for solving integer programming problems that works by solving a series of linear programming relaxations. At each stage, *cuts*, that is, linear inequalities that are satisfied by all integer feasible solutions but not by the non-integer optimal solution to the current relaxation, are added to the linear optimisation problem; this makes the relaxation more closely approximate the optimal solution of the original integer programming problem. This process is iterated until finding 'good' cuts becomes hard, at which point the current problem is split into two subproblems, to which branch & cut is applied recursively. For many problems, such as the Travelling Salesman Problem, branch & cut algorithms are currently among the best-performing complete search algorithms.

**Candidate Solution:** Element of the search space of a given combinatorial problem; typically a grouping, ordering or assignment of solution components. For many combinatorial problems, there is a natural distinction between *partial candidate solutions*, which may be further extended with additional solution components (without violating the fundamental structure of a candidate solution for the given problem), and *complete candidate solutions*, for which this is not the case. For example, in the case of the Satisfiability Problem, where natural solution components are assignments of truth values to individual propositional variables, partial candidate solutions may assign values to only some of the variables in the given formula, while complete candidate solutions specify truth values for all variables. Candidate solutions are also often called search positions; other terms that are sometimes used in the literature are *(search) state* and *(search) configuration*.

**Casanova:** High-performance stochastic local search algorithm for the Combinatorial Auction Winner Determination Problem; shares features with Novelty$^+$, a WalkSAT algorithm for the Satisfiability Problem, and with the Min-Conflicts Heuristic for the Constraint Satisfaction Problem.

**Chained Lin-Kernighan (CLK):** Iterated local search algorithm for the Travelling Salesman Problem.

**Chained Local Optimisation:** This term is sometimes used to refer to Iterated Local Search.

**Clause:** Disjunction of *literals* (i.e., of propositional variables or their negations) in a CNF formula. A clause is satisfied if, and only if, at least one of its literals is evaluated to true.

**Clause Penalty:** Numerical value associated with a clause in an instance of the Satisfiability Problem or the Maximum Satisfiability Problem. Clause penalties are used by dynamic local search algorithms and intuitively reflect the relative importance of ensuring the satisfaction of the respective clauses; they are often also referred to as a *penalty weights* or *clause weights*.

**Clause Weight:** Numerical value associated with a clause in an instance of the Weighted Maximum Satisfiability Problem (MAX-SAT) that reflects the relative importance of satisfying the respective clause. The term is also sometimes used to refer to clause penalties; but unlike the clause penalties used in dynamic local search algorithms for Weighted MAX-SAT, clause weights are an integral part of a given problem instance and are not modified during the search.

**CNF Formula:** Propositional formula in conjunctive normal form, that is, a disjunction of CNF clauses, where each CNF clause is a disjunction of *literals* (a literal is a propositional variables or its negation).

**Combinatorial Auction:** Auction mechanism that allows bidders to place bids for bundles of items. Combinatorial auctions have many practical applications. In contrast to conventional auctions, many basic problems associated with combinatorial auctions, such as the Combinatorial Auctions Winner Determination Problem, are computationally hard.

**Combinatorial Auction Winner Determination Problem (CAWDP):** Combinatorial problem in which the objective is to determine a feasible set of winning bids in a combinatorial auction such that the auctioneer's revenue is maximised. The CAWDP is an $\mathcal{NP}$-hard combinatorial optimisation problem that does not have an efficient approximation algorithm. Stochastic local search algorithms such as Casanova are among the state-of-the-art methods for solving this problem.

**Combinatorial Problem:** Problem in which given a set of *solution components*, the objective is to find a combination of these components with certain properties. For *combinatorial decision problems*, such as the Satisfiability Problem (SAT), the desired properties are stated in the form of logical conditions, while in the case of *combinatorial optimisation problems*, such as the Travelling Salesman Problem (TSP), the desired properties mainly consist of optimisation objectives which may be accompanied by additional logical conditions. Many (but not all) combinatorial problems are computationally hard and are solved by searching exponentially large spaces of candidate solutions.

**Complete Search Algorithm:** Search algorithm that, given sufficiently high run-time, is guaranteed to either find a solution of a given problem instance, or, if the instance is insoluble, to determine this fact with certainty. Complete optimisation algorithms, if run sufficiently long, are guaranteed to find solutions with provably optimal objective function values for any given problem instance. All systematic search algorithms are complete, while most stochastic local search algorithms are incomplete.

**Completion Time:** Time at which an operation or a job finishes processing in a scheduling problem.

**Constraint:** Relation between the values certain variables are allowed to take in a given instance of the Constraint Satisfaction Problem. The term is also used to refer to a logical condition on the properties of a candidate solution or solution component for a given combinatorial problem. *Hard constraints* represent conditions that any candidate solution needs to satisfy in order to be considered useful in a given application context, while *soft constraints* capture optimisation goals, not all of which may be satisfiable simultaneously.

**Constraint Satisfaction Problem (CSP):** Prominent combinatorial decision problem in artificial intelligence; given a set of variables and a set of relations (*constraints*) between these variables, the objective of the decision variant is to decide whether there exists an assignment of values to variables such that all constraints are simultaneously satisfied — such an assignment is called a *satisfying tuple*. *Finite discrete CSP*, the special case in which all variables have finite, discrete domains, is a widely studied $\mathcal{NP}$-complete problem of high theoretical and practical interest.

**Constructive Search:** Search paradigm in which the search process starts from an empty candidate solution and iteratively adds solution components until a complete candidate solution has been obtained. Constructive search algorithms are also known as *construction heuristics*. Constructive search can be seen as a special form of local search in a space of partial candidate solutions. A typical example for a constructive search algorithm is the Nearest Neighbour Heuristic for the Travelling Salesman Problem.

**Co-operative GLSM Model:** Extension of the Generalised Local Search Machine (GLSM) model that captures the simultaneous application of multiple search processes to solving a given problem instance; each of these search processes is represented by an individual GLSM. Co-operative GLSMs can be classified into *homogeneous co-operative GLSMs*, in which all individual GLSMs are identical, and *heterogeneous co-operative GLSMs*, which are comprised of different types of individual GLSMs. Furthermore, a distinction is made between *co-operative GLSMs without communication* and *co-operative GLSMs with communication*; while in the former case, the individual GLSMs do not communicate during the search, in the latter case, they exchange information, for instance, by means of a shared memory or a message passing mechanism. Co-operative GLSMs can be used to adequately represent various types of stochastic local search methods, including population-based SLS methods and algorithm portfolios.

**Correlation Length:** Measure of the ruggedness of a search landscape; typically determined empirically based on the autocorrelation function of an uninformed random walk in the landscape. Intuitively, smoother landscapes have larger correlation lengths. The correlation length is closely related to the autocorrelation coefficient and has a similar intuitive interpretation.

**CPU Time:** Measure for the run-time of an algorithm that is based on the actual computation time used by the respective implementation (process) on a given processor, typically measured in seconds, where one CPU second corresponds to one second of wall-clock time during which the CPU executes the given process only. CPU time is (mostly) independent of other processes running at the same time on a multi-tasking operating system, but depends on the processor type (model, speed, cache size) and may be affected by other aspects of the execution environment, such as main memory, compiler and operating system characteristics. When reporting run-times in CPU seconds, at least the processor model and speed, the amount of RAM, as well as the operating system type and version should be specified.

**Critical Block:** Sequence of critical operations that are assigned to the same machine in a candidate solution of a given instance of a scheduling problem.

**Critical Operation:** Operation in a candidate solution $s$ of a scheduling problem instance that cannot be delayed without increasing the makespan of $s$.

**Critical Path:** Maximum length sequence of critical operations in a candidate solution of a scheduling problem instance; a critical path can contain jobs that are assigned to different machines.

**Crossover:** Type of recombination mechanism that is based on assembling pieces or fragments from a linear representation of two or more candidate solutions (*parents*) into one or more new candidate solutions (*offspring*). Crossover is used in many Evolutionary Algorithms; sometimes, the term is used broadly to refer to any type of recombination mechanism.

**Cutoff Time:** Fixed time limit after which a run of a given algorithm is terminated, typically specified in terms of CPU time; in the context of restart mechanisms, the term is also used to refer to the (maximal) time interval after which the respective search process is restarted.

**Decision Problem:** Computational problem in which given a problem instance, the objective is to decide whether it satisfies a certain property. A prominent example is the Satisfiability Problem — the problem of deciding whether there exists an assignment of truth values to the variables in a given propositional formula such that the formula evaluates to true.

**Degenerate Distribution:** Fundamental type of probability distribution that characterises a random variable with only a single possible value. The cumulative distribution function of a degenerate distribution is a step function of the form $\theta[v](x) = 0$ if $x < v$ and 1 otherwise, where $v$ is the single value of the corresponding random variable. (This type of distribution is sometimes also called 'Dirac delta distribution'.)

**Discrete Lagrangian Method (DLM):** Family of dynamic local search algorithms that comprises several high-performance algorithms for the Satisfiability Problem and the Maximum Satisfiability Problem. DLM was originally motivated by Lagrangian methods for solving continuous optimisation problems.

**Dispatching Rule:** Heuristic function used in the context of a constructive search algorithm for scheduling problems for choosing in each construction step the operation or job to be scheduled next; often synonymously referred to as *priority rule*. An example for a well-known dispatching rule is the *Earliest Due Date (EDD) rule*, which always selects the unscheduled job with the earliest due date.

**Diversification:** Important property of a search process; *diversification mechanisms* help to ensure sufficiently strong exploration of the search space in order to avoid search stagnation and entrapment of the search process in regions of the search space that do not contain (sufficiently high-quality) candidate solutions. An important issue in the design of stochastic local search algorithms is to achieve a good (problem-specific) balance between diversification and intensification of the search process.

**DNA Code Design Problem (DNA-CDP):** Combinatorial optimisation problem with applications in biomolecular computation as well as in the design of DNA microarrays. The objective is to find a set of DNA words, that is, of strings over the alphabet $\{A, C, G, T\}$, that satisfy given constraints. Such a set is called a *DNA code*; typical constraints involve the Hamming distance between words or between words and their reverse complements, or the GC content of DNA words. The DNA Code Design Problem is related to well-known problems in coding theory. Although its theoretical complexity is still unknown, the DNA Code Design Problem appears to be computationally hard in practice.

**DNA Word:** Representation of a short strand of deoxyribonucleic acid (DNA) in the form of a string over the alphabet $\{A, C, G, T\}$. The DNA Code Design Problem involves finding sets of DNA words with certain properties.

**Domain:** In the sense of *variable domain*: set of values a given variable can take (e.g., in the Constraint Satisfaction Problem); in the sense of *problem/application domain*: particular type of problem or area of application.

**Double-Bridge Move:** Specific type of search step in the 4-exchange neighbourhood on edges in the Travelling Salesman Problem (TSP) that cannot be easily reversed by a short sequence of steps in the 2-exchange neighbourhood as performed, for example, by the Lin-Kernighan Algorithm; because of this latter property, double-bridge moves are often used in the perturbation phase of iterated local search algorithms for the TSP.

**Due Date:** Deadline for the completion of a job in a scheduling problem. Scheduling problems with due dates often use the (weighted) number of the jobs that are finished after their respective due dates or the sum of the weighted delays as an objective function.

**Dynamic Local Search (DLS):** Stochastic local search method that modifies the evaluation function of a subsidiary local search algorithm during the search. This is typically realised by associating *penalty weights* with components of a candidate solution; these weights are updated whenever the subsidiary local search procedure encounters a local minimum of the current evaluation function. Specific DLS algorithms differ primarily in their subsidiary local search and weight update procedures.

**Dynasearch:** Iterative improvement method that builds complex search steps based on optimal combinations of simple, independent search steps; search steps are considered *independent* if, and only if, they do not interfere with each other with respect to their effect on the evaluation function value and the feasibility of candidate solutions. Optimal combinations of simple, independent search steps are identified exploiting an algorithmic technique called *dynamic programming* (hence the name *Dyna*search).

**Essentially Incomplete:** Property of a Las Vegas algorithm; an algorithm is essentially incomplete if, and only if, it is not probabilistically approximately complete. Essentially incomplete SLS algorithms can get permanently trapped in non-solution regions of a given search space.

**Evaluation Function:** Function that assigns numerical values, typically real numbers greater than or equal to zero, to the candidate solutions of a given problem instance; most stochastic local search algorithms use an evaluation function for guiding the search process. In the case of SLS algorithms for optimisation problems, the evaluation function is often, but not always, identical to the objective function of the respective problem instance.

**Evolutionary Algorithms (EAs):** Large and diverse class of population-based SLS algorithms that is inspired by the process of biological evolution through selection, mutation and recombination. Evolutionary Algorithms are iterative algorithms that start with an initial population of candidate solutions and then repeatedly apply a series of the *genetic operators* selection, mutation and recombination. Using these operators, in each iteration of an EA, the current population is (completely or partially) replaced by a new set of *individuals*, that is, candidate solutions. Historically, several types of EAs can be distinguished: *genetic algorithms*, *evolutionary programming methods* and *evolution strategies*; but more recently, the differences between these types of EAs are becoming increasingly blurred. Many high-performance EAs for combinatorial problems use a subsidiary local search procedure for improving candidate solutions in each iteration of the search process; in many cases, these hybrid algorithms are based on genetic algorithms and are hence referred to as *genetic local search methods*. Slightly more generally, EAs that use subsidiary local search procedures are also known as *memetic search methods* or *memetic algorithms*.

**Exact (Optimisation) Algorithm:** Complete search algorithm for an optimisation problem, that is, an algorithm for solving an optimisation problem that is guaranteed to find an optimal solution for any given problem instance in time bounded by a function of the instance size.

**Exit:** Search position on the border of a given plateau that has a direct neighbour at a lower level; plateaus without exits, so-called *closed plateaus*, tend to make a problem instance difficult to solve for stochastic local search algorithms.

**Exponential Distribution:** Fundamental type of probability distribution with a cumulative distribution function of the form $ed[m](x) := 1 - 2^{-x/m}$ (where $m$ is the median of the distribution) or, equivalently, $Exp[\lambda](x) := 1 - e^{-\lambda \cdot x}$. Some prominent stochastic local search algorithms, such as GSAT with Random Walk or Novelty$^+$, show run-time distributions that closely resemble exponential distributions; SLS algorithms with exponential RTDs are not affected by static restart mechanisms and achieve optimal parallelisation speedup under multiple independent runs parallelisation.

**Exponentiated Subgradient Algorithm (ESG):** Dynamic local search method that uses *multiplicative penalty updates* whenever its underlying iterative best improvement algorithm encounters a local minimum of the given evaluation function; at the same time, a *smoothing mechanism* is applied to all penalty values. ESG has been motivated by subgradient optimisation methods for Lagrangian relaxation. The general framework has given rise to high-performance stochastic local search algorithms for the Satisfiability Problem and the Combinatorial Auctions Winner Determination Problem.

**Feasible Candidate Solution:** Alternative term for solution, that is, for a candidate solution that is an element of the solution set of a given stochastic local search algorithm and problem instance.

**Fitness:** Alternative term for the evaluation function value of a given candidate solution; originating from and often used in the context of Evolutionary Algorithms. Similarly, the term *fitness landscape* is often used synonymously with search landscape.

**Fitness-Distance Analysis:** A type of search space analysis in which the correlation between the fitness (i.e., evaluation function value) of candidate solutions and their distance to the next (optimal) solution is studied; this correlation is also known as *fitness-distance correlation (FDC)*. FDC can be measured by the *fitness-distance correlation coefficient* and graphically illustrated or analysed by means of *fitness-distance plots*.

**Flow Shop Scheduling Problem (FSP):** Scheduling problem in which each job consists of a number of atomic operations that are to be performed by different machines. In the FSP, all jobs have to pass through the machines in the same machine order. In the *Permutation FSP*, all jobs are processed in the same order on all machines, that is, a *candidate solution* is uniquely defined by a permutation of the jobs. The FSP is a special case of the Group Shop Scheduling Problem (GSP).

**Gantt Chart:** A graphical representation of a schedule in which the horizontal axis represents time, while different machines are distinguished along the vertical axis. Each operation is represented by a box that indicates when and where it is scheduled to be processed; the start and finish time of the operation correspond to the left and right boundaries of the respective box. Colours, shadings or labels are often used to indicate the grouping of operations into jobs or other properties, such as completion of a given operation after the due date of the respective job.

**Generalised Local Search Machine (GLSM):** Formal model of a hybrid SLS method that explicitly represents the search control in the form of a finite state machine whose *states* correspond to simple search strategies, while *transitions* between the states correspond to switches between search strategies. GLSM representations are often useful for designing, conceptualising and analysing complex stochastic local search algorithms.

**Genetic Algorithm (GA):** A type of Evolutionary Algorithm in which candidate solutions are represented typically by vectors of integers. While historically, candidate solutions were represented as bit strings, that is, by vectors of binary values $0$ and $1$, it is nowadays acknowledged that specialised representations (such as permutations of the integers $1, \ldots, n$ — which, for instance, are used to represent tours for the Travelling Salesman Problem) can be essential for the performance of GAs.

**Graph:** Mathematical structure consisting of a set of *vertices* and a set of *edges*, where each edge connects two vertices. An edge $e$ is said to be *incident* to a vertex $v$ if, and only if, $v$ is one of the vertices connected by $e$. In a *directed graph*, the edges are oriented, such that an edge from $v$ to $v'$, represented by $(v, v')$, is different from an edge from $v'$ to $v$, represented by $(v', v)$. In an *undirected graph*, the edges have no orientation, such that an edge between two vertices $v, v'$ connects $v$ with $v'$ as well as $v'$ with $v$; these undirected edges are often represented as sets $\{v, v'\}$. In an *edge-weighted graph*, a numerical value called an *edge weight* is associated with every edge; these weights can, for example, represent the length or cost of the respective edges.

**Graph Colouring Problem (GCP):** Well-known combinatorial problem in which the objective is to colour the vertices of a given graph in such a way that any two vertices connected by an edge are assigned different colours. Such an assignment of colours to vertices is called a *$k$-colouring* if, and only if, it uses at most $k$ different colours. The GCP is an $\mathcal{NP}$-hard problem with many applications, and stochastic local search algorithms are among the state-of-the-art methods for solving hard GCP instances.

**Greedy Randomised Adaptive Search Procedures (GRASP):** Class of stochastic local search algorithms that uses randomised greedy constructive search heuristics to generate a large number of different high-quality candidate solutions that are then further improved by a subsidiary local search procedure. These cycles of construction and local search phases are iterated until a termination condition is satisfied. The construction process in GRASP is called 'adaptive' because the heuristic value of a solution component in the constructive search typically depends on the components that are already present in the current partial candidate solution.

**Group Shop Scheduling Problem (GSP):** Scheduling problem that can be seen as a generalisation of the Job Shop, Open Shop and Flow Shop Scheduling Problems. In the GSP, each job consists of a number of atomic operations that are to be performed by different machines. The operations of each job are partitioned into a number of *groups*, and a total ordering of the groups of each job is given, while within each group the operations can be processed in any order.

**GSAT:** Simple stochastic local search algorithm for the Satisfiability Problem (SAT); essentially an iterative best improvement algorithm which in each search step flips (i.e., changes) the truth value assigned to one propositional variable. GSAT is also sometimes used to refer to the broader family of GSAT algorithms, which also includes variants of the basic GSAT algorithm, such as GSAT/Tabu or GSAT with Random Walk.

**GSAT/Tabu:** Simple stochastic local search algorithm for the Satisfiability Problem (SAT); essentially a simple tabu search variant of GSAT, in which whenever a variable is flipped it is declared tabu for a constant number of subsequent search steps.

**GSAT with Random Walk (GWSAT):** Prominent stochastic local search algorithm for the Satisfiability Problem (SAT); essentially a randomised iterative improvement algorithm. In each search step, the variable to be flipped is either selected as in GSAT or uniformly at random from the set of variables that occur in currently unsatisfied clauses. GWSAT can be seen as a simple hybrid combination of basic GSAT and a simple *conflict directed random walk* algorithm.

**Heuristic:** Function used for making certain decisions within an algorithm; in the context of search algorithms, typically used for guiding the search process, for example, by selecting solution components to be added to or modified in the current candidate solution. The term 'heuristic' is also commonly used to refer to algorithms that use heuristic functions and do not have certain performance guarantees, in particular w.r.t. finding solutions or reaching specific solution quality bounds.

**Hybrid SLS Method:** Stochastic local search method that combines different subsidiary SLS strategies with the goal to improve search performance or robustness. One of the simplest examples is Randomised Iterative Improvement, which probabilistically combines standard Iterative Improvement and Uninformed Random Walk. Hybrid SLS methods can often be modelled in an intuitive and representationally adequate way as Generalised Local Search Machines.

**Incomplete Search Algorithm:** Search algorithm that is not complete; incomplete algorithms may find (optimal) solutions to a given soluble problem instance, but generally cannot be guaranteed to do so, even if arbitrary (finite) amounts of run-time are allowed. Furthermore, incomplete algorithms generally cannot determine the insolubility of a given problem instance.

**Incumbent Candidate Solution:** Highest quality candidate solution encountered during the run of a stochastic local search algorithm for an optimisation problem, where solution quality is measured by the given objective function; often also referred to as *incumbent solution*. The term 'incumbent solution' is sometimes also used to refer to candidate solutions with the best evaluation function value encountered during a run of an SLS algorithm for a decision problem.

**Initialisation:** Elementary operation of a stochastic local search algorithm that is performed at the beginning of the search process and involves selecting a candidate solution from which the search is started. In SLS methods that use restart mechanisms, the same operation is also used for restarts during the search process.

**Initialising State:** State of a Generalised Local Search Machine that is used for search initialisation or restart; the search position after one search step performed in an initialising state is probabilistically independent of the position before the step.

**Insertion Neighbourhood:** Neighbourhood relation on permutations (or orderings) of solution components under which two permutations are direct neighbours if, and only if, one can be obtained from the other by removing an element from one position and inserting it at another position. The insertion neighbourhood is often used in scheduling problems; it is sometimes also called *shift neighbourhood*.

**Insoluble:** Property of a problem instance; a problem instance is called insoluble if, and only if, it has an empty solution set, and soluble otherwise. The term is typically applied to instances of decision problems, but also applies to optimisation problems whose instances may have empty solution sets.

**Integer Programming:** A set of methods for solving integer programming problems which comprises techniques such as branch & bound, branch & cut and dynamic programming.

**Integer Linear Programming (ILP) Problem:** Special case of the Integer Programming Problem in which the objective function as well as all feasibility constraints are linear functions of the decision variables. The set of techniques for solving ILP problems is called *Integer Linear Programming*. The special case in which the decision variables are restricted to the domain $\{0, 1\}$ is known as *0-1 Integer Linear Programming Problem* or *(Overconstrained) Pseudo-Boolean CSP*.

**Integer Programming (IP) Problem:** Combinatorial problem in which, given a set of numerical *decision variables*, a set of *feasibility constraints* and an optimisation objective, the goal is to find an optimal assignment of integer values to the variables such that all feasibility constraints are satisfied. Many combinatorial problems, such as the Set Covering Problem, can be represented as IP problems in an intuitive and straightforward way. The generalisation of IP in which some variables are not restricted to integer values is known as the *Mixed Integer Programming (MIP) Problem*. IP problems in which the variables are restricted to binary values 0 and 1 are often called *0-1 integer programming problems* or *Boolean programming problems*.

**Intensification:** Important property of a search process; *intensification mechanisms* aim to carefully examine a specific area of the given search space in order to find a solution or higher-quality candidate solutions. Intensification strategies are often strongly based on greedy heuristic guidance mechanisms. In high-performance stochastic local search algorithms, intensification strategies are often complemented with suitable diversification mechanisms.

**Interchange Neighbourhood:** Neighbourhood relation on permutations (or orderings) of solution components under which two permutations are direct neighbours if, and only if, one can be obtained from the other by interchanging the elements at two different positions. This neighbourhood relation is often used in scheduling problems but also in assignment problems, such as the Quadratic Assignment Problem; it is occasionally also called *swap neighbourhood*.

**Iterated Greedy (IG):** Stochastic local search method that can be seen as a variant of Iterated Local Search (ILS) in which *construction* and *destruction phases* are used instead of local search and perturbation phases. During each *construction phase*, a constructive search procedure is used to extend a partial candidate solution into a complete candidate solution. During the *destruction phase*, solution components are removed from the given complete candidate solution. Like ILS, IG uses an *acceptance criterion* to decide whether the search is continued from the new candidate solution $s'$ obtained from a given candidate solution $s$ by desctruction and subsequent construction, or whether $s'$ is abandoned and the search is continued from $s$. IG algorithms are among the state-of-the-art methods for solving the Set Covering Problem.

**Iterated Lin-Kernighan (ILK):** Iterated local search algorithm for the Travelling Salesman Problem that uses the Lin-Kernighan (LK) algorithm or a variant thereof as its subsidiary local search procedure; the perturbation procedure of ILK is based on random double-bridge moves, and its acceptance criterion only accepts new locally optimal candidate solutions that represent an improvement in the incumbent candidate solution.

**Iterated Local Search (ILS):** Stochastic local search method that is based on the repeated application of alternating *local search* and *perturbation phases*. Starting from an initial candidate solution, a *subsidiary local search procedure* is applied until a local minimum w.r.t. a given evaluation function is reached. Then, the following sequence of steps is iterated until a termination condition is satisfied. First, a *perturbation mechanism* is applied to the current candidate solution $s$, resulting in an intermediate candidate solution $s'$, which is typically not locally optimal. Next, *local search* is applied to $s'$, which leads to a locally optimal candidate solution $s''$. Finally, an *acceptance criterion* is used to decide from which candidate solution (typically either $s''$ or $s$) the search is continued. ILS algorithms can be seen as performing a biased walk in the space of local minima w.r.t. to a given evaluation function. Iterated local search algorithms are also known under various other names, including *Chained Local Optimisation* and *Large Step Markov Chains*.

**Iterated Robust Tabu Search (IRoTS):** Iterated local search algorithm for the Maximum Satisfiability Problem (MAX-SAT) whose subsidiary local search and perturbation phases are both based on Robust Tabu Search; furthermore, IRoTS uses a randomised acceptance criterion that is biased towards better-quality candidate solutions.

**Iterative Improvement:** Stochastic local search method that in each search step achieves an improvement in the evaluation function value of the current candidate solution. Depending on the pivoting rule that is used for selecting the search steps, different types of iterative improvement algorithms can be distinguished: *Iterative best improvement* algorithms always select a candidate solution with best function value within the current neighbourhood, while *iterative first improvement* algorithms check the neighbourhood in a given order and perform the search step that corresponds to the first improving candidate solution encountered during this check. Iterative improvement algorithms terminate when they encounter a local minimum of the given evaluation function.

**Job Shop Scheduling Problem (JSP):** Scheduling problem in which each job consists of a number of atomic operations that are to be performed by different machines; there is a total ordering of all the operations within each job, that is, the operations have to be performed in a given, fixed order. The JSP is a special case of the Group Shop Scheduling Problem (GSP).

**Large Step Markov Chains:** Iterated local search method that typically uses the Metropolis acceptance criterion. The term is occasionally used more broadly to refer to the general SLS method of Iterated Local Search.

**Las Vegas algorithm (LVA):** Stochastic algorithm that is guaranteed to only return correct solutions, but may run arbitrarily long without finding a solution for a given, soluble problem instance. The run-time of a Las Vegas algorithm on a given problem instance is characterised by a random variable. Stochastic local search algorithms form an important subclass of Las Vegas algorithms.

**Lateness:** Difference between the completion time of a job in a scheduling problem and its due date under a given schedule. Note that jobs that are completed before their due date have negative lateness.

**Level:** Alternative term for the evaluation function value of a candidate solution; mostly used in the context of search space analysis.

**Lin-Kernighan (LK) Algorithm:** Variable depth search algorithm for the Travelling Salesman Problem (TSP), also known as *Lin-Kernighan Heuristic*. Variants of this local search algorithm form the basis of most current state-of-the-art algorithms for the TSP. Historically, LK was one of the first variable depth search algorithms.

**Local Minima Density:** Relative frequency of local minima within a given search space; analogous to the solution density, the local minima density of a search space of size $\#S$ with $l$ local minima is defined as $l/\#S$. Problem instances with higher local minima density tend to be easier to solve for stochastic local search algorithms. Local minima density is closely related to the ruggedness of a given search landscape and can sometimes be estimated using ruggedness measures, such as correlation length.

**Local Minimum:** Candidate solution whose evaluation function value is smaller than or equal to the evaluation function values of any of the candidate solutions in its neighbourhood; a candidate solution is a *strict local minimum* if, and only if, all of its neighbours have strictly larger evaluation function values.

**Local Search:** Algorithmic method for searching a given space of candidate solutions that starts from an initial candidate solution and then iteratively moves from one candidate solution to a candidate solution from its direct neighbourhood, based on local information, until a termination condition is satisfied.

**Log-Log Plot:** Graphical representation in which both axes of a given coordinate system are shown in logarithmic scale. Log-log plots are particularly useful in the context of scaling and correlation analyses, since the graphs of polynomial functions appear as straight lines in such plots; they are also often used when analysing the tails of distributions, such as run-time distributions.

**Makespan:** Commonly used objective function for scheduling problems that measures the maximum completion time of any job under a given candidate schedule.

**Many-Valued Satisfiability Problem (MV-SAT):** Generalisation of the Satisfiability Problem (SAT), in which variables can have domains with more than two values. An instance of MV-SAT is given by a generalisation of a CNF formula in which each literal specifies or rules out one or more values of a many-valued variable. MV-SAT is a special case of the Constraint Satisfaction Problem (CSP). Like CSP and SAT, MV-SAT is $\mathcal{NP}$-complete. MV-SAT is also known as *Multi-Valued SAT*.

**Maximum Constraint Satisfaction Problem (MAX-CSP):** Generalisation of the Constraint Satisfaction Problem (CSP) in which, given a CSP instance, the objective is to find a variable assignment that maximises the number of satisfied constraints. In *Weighted MAX-CSP*, weights associated with each constraint can be used to prioritise the satisfaction of constraints, and the objective is to find an assignment that maximises the total weight of the satisfied constraints.

**Maximum Satisfiability Problem (MAX-SAT):** Generalisation of the Satisfiability Problem in which, given a CNF formula, the objective is to find a variable assignment that maximises the number of satisfied clauses. In *Weighted MAX-SAT*, weights associated with each clause can be used to prioritise the satisfaction of clauses, and the objective is to find an assignment that maximises the total weight of the satisfied clauses.

**$\mathcal{MAX}$–$\mathcal{MIN}$ Ant System ($\mathcal{MM}$AS):** Class of Ant Colony Optimisation algorithms that uses limits on the feasible values of the pheromone trails as well as additional intensification and diversification mechanisms. $\mathcal{MM}$AS forms the basis for many successful applications of ACO algorithms to $\mathcal{NP}$-hard optimisation problems. Unlike many other ACO algorithms, as a result of its use of limits on the pheromone trails, $\mathcal{MM}$AS is provably probabilistically approximately complete (PAC).

**Memetic Algorithms (MAs):** Class of population-based SLS algorithms that combines the evolutionary operators mutation, recombination and selection with a subsidiary local search procedure or, more generally, problem-specific heuristic information. MAs that use subsidiary local search can be seen as evolutionary algorithms that search a space of locally optimal candidate solutions.

**Metaheuristic:** Generic technique or approach that is used to guide or control an underlying problem-specific heuristic method (for example a local search algorithm or a constructive search algorithm) in order to improve its performance or robustness. The term is also widely used to refer to simple, hybrid and population-based SLS methods, such as Simulated Annealing, Tabu Search, Iterated Local Search, Evolutionary Algorithms and Ant Colony Optimisation.

**Metric TSP:** Special case of the Travelling Salesman Problem (TSP), in which the vertices of the given graph correspond to points in a metric space, and the edge weights correspond to metric distances between pairs of points. A prominent special case is the *Euclidean TSP*, which is based on the standard Euclidean metric.

**Metropolis Acceptance Criterion:** Widely used acceptance criterion that deterministically accepts a new candidate solution $s'$ if it has a better evaluation function value than the current candidate solution $s$; otherwise, $s'$ is accepted with probability $e^{-|g(s)-g(s')|/T}$, where $g(s)$ and $g(s')$ are the evaluation function values of $s$ and $s'$, respectively, and $T$ is a parameter called *temperature*. This acceptance criterion is frequently used in simulated annealing algorithms and can also be found in a number of iterated local search algorithms.

**Min-Conflicts Heuristic (MCH):** Well-known stochastic local search algorithm for the finite discrete Constraint Satisfaction Problem. MCH is basically an iterative improvement method, which in each search step first selects a variable that appears in a currently unsatisfied constraint, and then assigns a value to this variable such that the number of unsatisfied constraints is maximally reduced, that is, minimised within the local neighbourhood.

**Multiple Independent Runs Parallelisation:** Conceptually simple form of parallelisation, in which multiple runs of a given stochastic algorithm applied to the same problem instance are executed in parallel and independently from each other. Multiple independent runs parallelisation typically only involves minimal communication between the respective processes (or no communication at all) and is very easy to implement. Most stochastic local search algorithms achieve high parallelisation speedup under this form of parallelisation.

**Mutation:** Basic operation used in Evolutionary Algorithms to introduce modifications to a member of the population, that is, to a candidate solution.

**Nearest Neighbour Heuristic:** Constructive search method for the Travelling Salesman Problem (TSP) which, starting at a randomly chosen vertex in the given graph, in each step follows an edge with minimal weight connecting the current vertex to one of the neighbouring vertices that have not yet been visited. The resulting *nearest neighbour tours* are typically of substantially lower quality than those obtained by commonly used stochastic local search algorithms for the TSP.

**Neighbourhood:** The set of candidate solutions that are direct neighbours of a given candidate solution (e.g., the current candidate solution) under a given neighbourhood relation. (The term is sometimes also used to refer to a neighbourhood relation.)

**Neighbourhood Graph:** Graph whose vertices are the candidate solutions of a given problem instance and whose edges connect candidate solutions that are direct neighbours of each other under the given neighbourhood relation. The search trajectory of any local search algorithm can be seen as a walk in the respective neighbourhood graph. The *diameter of the neighbourhood graph*, defined as the maximal distance between any pair of vertices $s$ and $s'$, where distance is measured as the minimal number of edges that need to be traversed to reach $s$ from $s'$, is an important concept in search space analysis.

**Neighbourhood Relation:** Important component of a local search algorithm; a binary relation that determines the direct neighbours of any candidate solution of a given problem instance, that is, the set of candidate solutions that can potentially be reached in a single search step.

**NK-Landscapes:** Statistical model of search landscapes with controllable ruggedness. The model has two main parameters: $N$ specifies a number of binary variables and $K$ indicates the number of other variables on which the evaluation function contribution of each variable depends. For fixed $N$, the ruggedness of an

NK-landscape increases with $K$. The problem of finding a global minimum in an NK-landscape is $\mathcal{NP}$-hard.

**Noise Parameter:** Parameter that controls the degree of randomness used within the search steps of a given stochastic local search algorithm. Examples for noise parameters are the walk probability in GSAT with Random Walk, or the tabu tenure in WalkSAT/Tabu.

**Non-Boolean Satisfiability Problem (NB-SAT):** Generalisation of the Satisfiability Problem (SAT). An instance of NB-SAT is given by a generalisation of a CNF formula in which each literal specifies or rules out precisely one value of the corresponding non-Boolean variable. NB-SAT is a special case of the Constraint Satisfaction Problem. Like SAT and CSP, NB-SAT is $\mathcal{NP}$-complete.

**Non-Delay Schedule:** A schedule under which no machine in a given scheduling problem is ever kept idle as long as there is an operation that can be processed.

**Non-Oblivious SLS Method:** Stochastic local search method that uses a *non-oblivious evaluation function*, that is, an evaluation function that captures the degree to which constraints of a given problem are satisfied or violated. In the case of the Satisfiability or Maximum Satisfiability Problem, the constraints are the clauses of the given CNF formula, and the degree of satisfaction of a clause $c$ under a given assignment $a$ is the number of literals in $c$ that are satisfied under $a$.

**Novelty$^+$:** Prominent high-performance stochastic local search algorithm for the Satisfiability Problem (SAT). Novelty$^+$ is a WalkSAT algorithm that uses a history-based, randomised greedy mechanism for selecting the propositional variable to be flipped in each search step. Through the use of an unconditional random walk mechanism, Novelty$^+$ is provably probabilistically approximately complete (PAC).

**$\mathcal{NP}$:** Computational complexity class; a decision problem is in $\mathcal{NP}$ if, and only if, it can be solved in polynomial time w.r.t. the size of a given problem instance by a *nondeterministic Turing machine* — an idealised, theoretical model of computation. Intuitively, a problem is in $\mathcal{NP}$ if there is a polynomial-time algorithm for checking the correctness of a solution to any given problem instance.

**$\mathcal{NP}$-complete:** Computational complexity property; a decision problem is $\mathcal{NP}$-complete if, and only if, it is in the complexity class $\mathcal{NP}$ and it is $\mathcal{NP}$-hard. Intuitively, $\mathcal{NP}$-complete problems are the hardest problems that can be solved in polynomial time w.r.t. the size of a given problem instance by a *nondeterministic Turing machine* — an idealised, theoretical machine model. It is strongly suspected (though yet unproven) that solving $\mathcal{NP}$-complete problems with any conventional, implementable algorithm (i.e., using a computational model equivalent to a deterministic Turing machine) takes time exponential in instance size.

**$\mathcal{NP}$-hard:** Computational complexity property; a problem is $\mathcal{NP}$-hard if, and only if, any algorithm that would solve it in polynomial time w.r.t. the size of a given instance could also be used to solve (suitably encoded instances of) any other problem in the complexity class $\mathcal{NP}$ in polynomial time. Unlike the notion of $\mathcal{NP}$-completeness, $\mathcal{NP}$-hardness applies to optimisation problems as well as to decision problems.

**Objective Function:** Important component of any combinatorial optimisation problem; a function that assigns a numerical value called solution quality (typically a real number greater than or equal to zero) to each candidate solution of a given problem instance. The objective function is the measure to be minimised or maximised in a combinatorial optimisation problem and typically models the cost or quality of a candidate solution.

**Oblivious SLS Method:** Stochastic local search method that uses an *oblivious evaluation function*, that is, an evaluation function that ignores the degree to which constraints of a given problem are satisfied or violated; most widely known, high-performance stochastic local search algorithms are of this type. (See also non-oblivious SLS methods.)

**Open Shop Scheduling Problem (OSP):** A scheduling problem in which each job consists of a number of atomic operations that are to be performed by different machines. In the OSP, there are no precedence constraints among the operations of each job, that is, the operations of each job can be processed in arbitrary order. The OSP is a special case of the Group Shop Scheduling Problem.

**Operation:** Basic action to be performed by a machine in a scheduling problem on a given job; depending on the type of scheduling problem, jobs may consist of one or more operations (single- *vs* multi-stage scheduling problems).

**Optimal Solution:** Solution of an optimisation problem whose objective function value is provably minimal within the entire search space of the given problem instance in case of a minimisation problem, or provably maximal in case of a maximisation problem. For many hard and large instances of optimisation problems, (provably) optimal solutions are not known, and empirically best or quasi-optimal solutions have to be used instead, for example, in the context of assessing the performance of stochastic local search algorithms.

**Optimisation Las Vegas Algorithm (OLVA):** Las Vegas algorithm for an optimisation problem for which, when applied to a given problem instance, the solution quality obtained at run-time $t$ is characterised by a random variable $SQ(t)$. The performance of an optimisation Las Vegas algorithm on a given problem instance is characterised by a bivariate run-time distribution. Stochastic local search algorithms for optimisation problems form an important subclass of the optimisation Las Vegas algorithms.

**Optimisation Problem:** Computational problem in which given a problem instance $\pi'$ and objective function $f$, the goal is to find a candidate solution of $\pi'$ that minimises (or maximises) $f$. A prominent example is the Travelling Salesman Problem — the problem of finding a round trip of minimal length in a given edge-weighted graph that visits every vertex exactly once.

**Parallel Machine Scheduling Problem:** Type of scheduling problem in which several machines are available for processing the given jobs; each job consists of a single operation that is performed by one machine. In *identical parallel machine problems*, the processing time of a job is independent of the machine on which it is processed; in *uniform parallel machine problems*, each machine has a speed that uniformly affects the processing times of all jobs assigned to it; and in *unrelated parallel machine problems*, the processing times of jobs may depend in a non-uniform way on the machines to which they are assigned.

**Parallelisation Speedup:** Speedup achieved by a parallel algorithm $A_p$ as compared to a functionally equivalent sequential algorithm $A_s$. Formally, parallelisation speedup is defined as the ratio of the run-time of $A_s$ (running on a single processor) and the parallel run-time of $A_p$ running on multiple processors, where one unit of parallel run-time consists of one unit of sequential run-time on each of the processors that are involved in the execution of the parallel algorithm $A_p$ at the given time. Parallelisation speedup depends on the given algorithms $A_p$ and $A_s$ as well as on the given input (i.e., problem instance).

**Path:** Sequence $p$ of vertices in a given graph $G$ such that any pair of successive vertices in $p$ are connected by an edge in $G$. In a *cyclic path* or *cycle in $G$*, the first and the last vertex in $p$ are identical. A *Hamiltonian path in $G$* contains every vertex in $G$ exactly once; a *Hamiltonian cycle in $G$* is a cyclic path $p$ that contains every vertex in $G$ exactly once with the exception of the first and last vertex in $p$. If $G$ is an edge-weighted graph, the *weight of a path $p$ in $G$* is the total weight of the edges in $p$.

**Peak Performance:** Maximal performance achieved by a given parameterised algorithm applied to a given problem instance (or set of problem instances) when using optimised parameter settings. Even though in many cases peak performance is only obtained for manually tuned, instance-specific parameter settings, it is a useful measure for assessing the *performance potential* of an algorithm.

**Permutation:** Ordering of a set of objects; formally, a permutation over a set $S$ with $n$ elements can be defined as a mapping of the integers $1, \ldots, n$ to elements of $S$ such that each integer is mapped to a unique element of $S$.

**Perturbative Search:** Search paradigm in which candidate solutions are iteratively perturbed by modifying one or more solution components in each search step. Typically, the candidate solutions used in a perturbative search algorithm are complete candidate solutions. Iterative improvement algorithms, such as 2-opt local search for the Travelling Salesman Problem, are typical examples of perturbative search methods.

**Pivoting Rule:** Rule that defines the mechanism for determining search steps in an iterative improvement algorithm. The most widely used pivoting rules are the so-called *best improvement* and *first improvement* strategies. *Best improvement* always selects the search step that leads to the neighbouring candidate solution with the best evaluation function value. *First improvement* searches the neighbourhood of the current candidate solution $s$ in a given order and selects the search step leading to the first neighbour with a strictly better evaluation function value than $s$ encountered during this process.

**Plateau:** Maximally connected set of search positions at the same level of a given search landscape; search spaces with extensive plateaus, as can be found for typical instances of the Satisfiability Problem (SAT), can be challenging for stochastic local search (SLS) algorithms, since plateaus are by definition regions in which the algorithm does not have any heuristic guidance. Plateaus with exits are called *open plateaus*, while plateaus without exits are referred to as *closed plateaus*; the latter consists entirely of local minima or strict local minima positions and tend to render problem instances difficult to solve for SLS algorithms.

**Plateau Connection Graph (PCG):** Representation of the plateau structure of a search landscape; the vertices of a PCG represent plateaus in the given landscape, and the edges correspond to exits connecting the respective plateaus. A PCG forms a complete partition of a given landscape, which can be seen as a refinement of the respective basin partition tree.

**Population-Based SLS Method:** Class of stochastic local search algorithms that maintain a *population*, that is, a set of candidate solutions of the given problem instance. In each search step, one or more elements of the population (i.e., individual candidate solutions) may be modified. The use of a population of candidate solutions often helps to achieve adequate diversification of the search process. Examples for population-based SLS algorithms include Evolutionary Algorithms, Memetic Algorithms and Ant Colony Optimisation.

**Precedence Constraint:** Constraint that indicates that a job (or operation) in a scheduling problem has to be executed before (but not necessarily immediately before) another job (or operation). Precedence constraints occur in many real-world scheduling problems.

**Probabilistically Approximately Complete (PAC):** Property of a stochastic search algorithm; an algorithm is PAC if, and only if, for increasingly long run-times the probability of finding a solution (or optimal solution) of any soluble problem instance gets arbitrarily close to one. Algorithms that are PAC can never get trapped in non-solution regions of the search space. However, they may still require a very long time to escape from such regions.

**Probabilistic Domination:** Performance relationship between two Las Vegas algorithms; LVA $A$ probabilistically dominates LVA $B$ if, and only if, for any given run-time, $A$ achieves at least as high a success probability as $B$, and there is at least one run-time value for which the success probability of $A$ exceeds that of $B$. For optimisation Las Vegas algorithms, probabilistic domination is defined analogously, based on the probability of reaching a given solution quality bound $q$ within the given time, and $A$ probabilistically dominates $B$ if, and only if, it probabilistically dominates $B$ for any given $q$.

**Probabilistic Iterative Improvement (PII):** SLS method that probabilistically accepts a neighbouring candidate solution $s'$ based on an acceptance criterion that takes into account the difference in evaluation function value of $s'$ and the current candidate solution, $s$. Iterative Improvement as well as Randomised Iterative Improvement can be seen as special cases of PII.

**Pseudo-Boolean CSP (PB-CSP):** Special case of the finite discrete Constraint Satisfaction Problem, also known as *Linear Pseudo-Boolean Programming*. In Pseudo-Boolean CSP, all variables have domains $\{0, 1\}$, and the constraints have the form of linear inequalities, such as $x_1 - 2x_2 + 3x_3 \geq 0$ (or other relations, including equalities and strict inequalities). *Overconstrained Pseudo-Boolean CSP (OPB-CSP)* is an optimisation variant of PB-CSP, in which in addition to a set of constraints as in PB-CSP (*hard constraints*) a set of *soft constraints* of the same general form is given, and the objective is to find an assignment that leaves a minimal number of soft constraints unsatisfied while satisfying all hard constraints. OPB-CSP can be seen as a special case of the 0-1 Integer Linear Programming (ILP) Problem.

**Quadratic Assignment Problem (QAP):** Combinatorial optimisation problem in which, given a number of *objects* and *locations*, *flow values* between the objects and *distances* between the locations, the goal is to assign all objects to different locations such that the sum of the products of the given flow values between pairs of objects and the distances between the respective locations is minimised. The QAP has various practical applications, such as facility layout and halftone rendering problems. The QAP is $\mathcal{NP}$-hard and is considered to be one of the empirically hardest combinatorial optimisation problems. Stochastic local search algorithms currently represent the only feasible approach for solving large QAP instances.

**Qualified Run-Time Distribution (QRTD):** Probability distribution of the time required by an optimisation Las Vegas algorithm to reach or exceed a given solution quality bound $b$ for a given problem instance; corresponds precisely to the run-time distribution required by a variant of the same algorithm that terminates as soon as the specified solution quality bound has been reached or exceeded, and hence solves the decision variant for bound $b$ associated with the given problem instance.

**Quantile:** Statistical measure; given a numerical random variable $X$, the $\alpha$-quantile of the respective probability distribution, denoted $q_\alpha(X)$, is defined as the minimal value $x'$ such that $P(X \leq x') \geq \alpha$. The $0.5$-quantile is the *median* of the distribution, and the $p/100$-quantiles are also known as the *p-th percentiles*. The medians of empirical distributions are often statistically more stable than the respective means and can be easily and intuitively read off a graphical representation of the cumulative distribution function. *Quantile ratios*, such as $q_{0.9}(X)/q_{0.1}(X)$, are measures of variation which, like the variation coefficient, are invariant w.r.t. multiplicative scaling of the underlying random variable. Quantiles and quantile ratios are often used in the empirical analysis of the performance and behaviour of Las Vegas algorithms.

**Quantile-Quantile Plot (QQ Plot):** 2-dimensional graphical representation of the correlation between the quantiles of two probability distributions, in which the two coordinates of each data point correspond to the same quantile of two given distributions. Quantile-quantile plots are often used to visually illustrate the differences and similarities between two distributions. They are also useful for informally testing whether a given empirical distribution, such as a run-time distribution or solution quality distribution, can be modelled by a theoretical distribution, such as an exponential or normal distribution. (Formally, this type of hypothesis can be tested using statistical hypothesis tests, such as the *Kolmogorov-Smirnov* or $\chi^2$ *goodness-of-fit tests*.)

**Quasi-Optimal Solution:** Solution of an optimisation problem that is believed or suspected to be an optimal solution, but whose optimality has not been proven. Quasi-optimal solutions are often obtained by long runs of high-performance stochastic local search (SLS) algorithms. (Such algorithms need to be able to find provably optimal solutions for instances that are small enough to allow the application of state-of-the-art complete search algorithms; ideally, they should also be probabilistically approximately complete.) Often, as substantial improvements are made in state-of-the-art complete algorithms for a problem, quasi-optimal solutions for well-known benchmark instances are verified to be optimal, while sometimes, they turn out to be suboptimal.

**Randomised Iterative Improvement (RII):** Hybrid SLS method in which with a probability $wp$, called walk probability or noise parameter, a neighbouring candidate solution is picked at random, while with a probability of $1 - wp$, a standard iterative improvement step is performed. Note that this mechanism allows for arbitrary long sequences of uninformed random walk steps, which renders RII provably probabilistically approximately complete.

**Reactive Tabu Search (RTS):** Tabu search method that dynamically adapts the tabu tenure during the search based on a limited amount of memory on previously visited search positions. Additionally, an escape mechanism is triggered whenever sufficient evidence for search stagnation has been gathered; this escape mechanism typically consists of a number of uninformed random walk steps that are executed starting from the current candidate solution.

**Recombination:** Fundamental operation in Evolutionary Algorithms or Memetic Algorithms that generates one or more new candidate solutions (*offspring*) by combining parts (solution components) of two or more existing candidate solutions (*parents*). In many cases, a special type of recombination called crossover is used.

**Release Date:** Earliest time at which a job in a scheduling problem is available for processing.

**Restart Mechanism:** Mechanism that restarts a search process from a new initial candidate solution under certain conditions. When using a *static restart mechanism*, reinitialisation occurs regularly after a fixed number of search steps, while *dynamic restart mechanisms* may restart the search if no improvement in the incumbent candidate solution has been achieved for a certain number of steps, or if the search process is trapped in a local minimum of the given evaluation function.

**Robust Tabu Search (RoTS):** Tabu search method that repeatedly chooses the value of the tabu tenure parameter uniformly at random from a given integer interval during the search; this often results in increased robustness of performance compared to standard Tabu Search, which uses a constant tabu tenure value. Many high-performance RoTS algorithms also use additional diversification mechanisms to prevent or overcome search stagnation.

**RTQ (Run-Time Over Solution Quality):** Development of a statistical measure of the run-time in dependence of the solution quality achieved by an optimisation Las Vegas algorithm; statistical measures commonly used in this context include the mean, median and quantiles of the qualified run-time distribution for the respective solution quality bound. Like SQTs, RTQs summarise the underlying bivariate run-time distributions and reflect the tradeoff between run-time and solution quality for the given algorithm on a specific problem instance. RTQs are closely related to SQTs but somewhat less intuitive; consequently, they are rarely used in the analysis of the behaviour of optimisation LVAs.

**Ruggedness:** Property of a search landscape that is closely related to its local minima density; intuitively, rugged landscapes have many local minima, and the respective problem instances tend to be harder to solve for stochastic local search algorithms. Landscape ruggedness is often measured by means of the autocorrelation coefficient or correlation length.

**Run-Time Distribution (RTD):** Probability distribution of the time required by a Las Vegas algorithm to solve a given problem instance. The RTDs for optimisation Las Vegas algorithms are bivariate distributions of run-time and solution quality. RTDs that are based on measurements of run-time in terms of elementary operations of an algorithm (such as search steps in the case of stochastic local search algorithms) are called *run-length distributions (RLDs)*. RTDs and RLDs play an important role in the empirical analysis of the behaviour and performance of SLS algorithms.

**Satisfiability Problem (SAT):** Prototypical combinatorial decision problem in which given a propositional formula $F$, the objective is to decide whether there is an assignment of truth values to the variables in $F$ such that $F$ becomes satisfied. Commonly, SAT is used to refer to a version of the general satisfiability problem that is restricted to CNF formulae. SAT (in general and for CNF formulae) is $\mathcal{NP}$-complete. Stochastic local search algorithms are among the best methods for solving certain types of hard SAT instances.

**Scaling:** Dependence of a property of an algorithm, such as run-time, on the size of the input, that is, the problem instance to be solved. Often, scaling is analysed for a specific family or distribution of problem instances. The term is also used occasionally to refer to the dependence of certain properties of the instances themselves on instance size. Finally, the term scaling is often applied to the operation of data renormalisation, for example, multiplication with a constant. *Finite size scaling* is a technique originally developed in statistical physics that is used for extrapolating ensemble properties of very large problem instances, such as the occurrence of a solubility phase transition, from empirical data obtained for small instance sizes.

**Scaling and Probabilistic Smoothing (SAPS):** Dynamic local search method that is closely related to the Exponentiated Subgradient Algorithm (ESG). Different from ESG, SAPS performs penalty smoothing probabilistically, which results in substantial performance improvements. *Reactive SAPS (RSAPS)* adaptively modifies the smoothing probability to achieve increased performance robustness. SAPS and RSAPS have been shown to reach state-of-the-art performance for certain types of instances of the Satisfiability Problem and the Maximum Satisfiability Problem.

**Scheduling Problems:** Large and important class of combinatorial problems in which given a set of *jobs* that have to be processed by a set of *machines*, the goal is to find an optimal *schedule*, that is, a mapping of jobs to machines and times at which they are processed, under a given *objective function* and subject to *feasibility constraints*. In *single-stage scheduling problems*, each job consists of one atomic operation that is executed by exactly one machine in a single processing stage, while in *multi-stage scheduling problems*, a job can consist of multiple operations that may have to be performed by different machines in multiple stages. Similarly, in a *single-machine scheduling problem*, only one machine is available to process the jobs, while in a *multi-machine scheduling problem*, several machines are available. Scheduling problems arise in many application areas; in many, but not all cases, they are $\mathcal{NP}$-hard. Stochastic local search algorithms are among the best methods for solving hard scheduling problems.

**Search Cost Distribution (SCD):** Probability distribution of *search cost* across a set of problem instances, where search cost is a measure of the run-time (or other resources) required by a given algorithm for solving a specific problem instance. For stochastic search algorithms, search cost is typically defined as the mean or a quantile of the run-time distribution for a given problem instance. In this case, the run-time distribution characterises the variation of run-time over multiple runs of the algorithm on the same problem instance, while a search cost distribution reflects the variation of run-time over a given set or distribution of problem instances. In the literature, SCDs are sometimes referred to as 'hardness distributions'.

**Search Landscape:** Mathematical structure comprised of the search space, the neighbourhood relation and an evaluation function for a given problem instance. Characteristics of this landscape, such as its solution density, local minima density or ruggedness, have a crucial impact on the behaviour of stochastic local search algorithms.

**Search Position:** Alternative term for a candidate solution of a given problem instance; often used in the context of search space analysis.

**Search Space:** Set of all candidate solutions of a given problem instance. The size of the search space, that is, the number of candidate solutions, typically scales at least exponentially w.r.t. instance size within a given family of problem instances, and it can strongly depend on the representation of candidate solutions. For example, for an instance of the Travelling Salesman Problem with $n$ vertices and $\Theta(n^2)$ edges, candidate solutions may be represented as cyclic permutations of the $n$ vertices or by $\Theta(n^2)$ binary variables, each of which indicates whether or not a specific edge is part of a given candidate solution; these two different representations lead to search spaces of size $\Theta(n!)$ and $\Theta(2^{(n^2)})$, respectively. The term *search space* is occasionally used more broadly to refer to a search landscape.

**Search Space Analysis:** Investigation of features and characteristics of the search space, or — more generally — the search landscape of a given problem instance. Search space analysis plays an important role in explaining, understanding and improving the behaviour of stochastic local search algorithms.

**Search Step:** Elementary operation of a stochastic local search algorithm, in which the search moves from a candidate solution $s$ to a candidate solution $s'$ in the direct neighbourhood of $s$. A search step typically involves the modification, addition or removal of one or more solution components.

**Search Trajectory:** Finite sequence of search positions (i.e., candidate solutions) as (possibly) visited in successive search steps of a given stochastic local search algorithm.

**Selection:** Fundamental operation in Evolutionary Algorithms or Memetic Algorithms that is used for selecting the individuals (i.e., candidate solutions) from the current population to be retained for the next iteration of the search process — these surviving individuals form the next generation population. More generally, selection mechanisms may also be used for choosing the individuals that undergo specific operations in a population-based SLS algorithm, such as mutation or recombination in the context of Evolutionary Algorithms, or local search in the context of Memetic Algorithms or Ant Colony Optimisation.

**Semi-Log Plot:** Graphical representation in which one of the two axes of a coordinate system is shown in logarithmic scale. Semi-log plots with a logarithmic $x$-axis are particularly useful in the context of scaling and correlation analyses, since the graphs of exponential functions appear as straight lines in such plots; they are also often used when analysing distributions, such as run-time distributions, whose shape in a semi-log plot with logarithmic $x$-axis appears invariant under multiplication by a constant factor.

**Sequencing Problems:** Class of scheduling problems in which the goal is to determine an optimal ordering for processing a given set of jobs. Different from a schedule, such an ordering or sequence does not assign time-slots to jobs (or operations); however, in many cases, a schedule can be easily derived from the solution of a sequencing problem. The term is also often used to refer to scheduling problems whose candidate solutions naturally correspond to sequences of jobs or operations. Examples of sequencing problems include the Permutation Flow Shop Scheduling Problem and the Single-Machine Total Weighted Tardiness Scheduling Problem.

**Set Covering Problem (SCP):** Well-known combinatorial optimisation problem in which, given a set $A$ of objects and a family $F$ of subsets of $A$, the goal is to find the minimum number of subsets from $F$ such that every element of $A$ occurs in at least one of the chosen subsets, that is, the set $A$ is *covered* by the selected sets. This problem is also known as the *Minimum SCP* or *Unicost SCP*. In the *Weighted SCP*, a weight is associated with each subset in $F$, and the objective is to find a selection of subsets from $F$ covering $A$ with minimum total weight. The SCP arises in various practical applications, such as crew scheduling in airline, railway and bus companies; stochastic local search algorithms are among the best methods for solving this $\mathcal{NP}$-hard optimisation problem.

**Simulated Annealing (SA):** Stochastic local search method that is inspired by the physical process of the annealing of solids. In each search step, a neighbour $s'$ of a candidate solution $s$ is generated by a *proposal mechanism*, and an *acceptance criterion* is used to probabilistically decide whether the search will be continued from $s$ or $s'$; this decision is made based on the evaluation function values of $s$ and $s'$ as well as on the value of a parameter $T$ called *temperature*, which is modified during the search process according to an annealing schedule. Many simulated annealing algorithms use the Metropolis acceptance criterion.

**Single-Machine Total Weighted Tardiness Scheduling Problem (SMTWTP):** Single-machine scheduling problem in which for each job a processing time, a due date and an importance weight are given; all jobs are available for processing at time zero, and the objective is to find a schedule that minimises the sum of the weighted tardiness values of the jobs. The SMTWTP is an $\mathcal{NP}$-hard optimisation problem. The special case in which all jobs have the same weight is known as the *Single-Machine Total Tardiness Scheduling Problem (SMTTP)*.

**Solubility Phase Transition:** Transition from one to zero in the probability that a randomly drawn instance from a parameterised family of problem instance distributions is soluble as some parameter is varied. A classical example is

Uniform Random $k$-SAT for a fixed number of variables and variable number of clauses. In many cases, a solubility phase transition is accompanied by qualitative changes in other properties of the instance distribution, in particular, the expected time for solving a given problem instance.

**Soluble:** Property of a problem instance; a problem instance is called *soluble* if, and only if, it has at least one solution, and *insoluble* otherwise. The term is typically applied to instances of decision problems; many optimisation problems are defined in such a way that their instances are always soluble in the sense that they have a non-empty set of (feasible) solutions.

**Solution:** Element of the solution set, $S'$, of a given stochastic local search (SLS) algorithm for a given problem instance. A solution of a combinatorial decision problem is typically defined as a candidate solution that satisfies certain logical conditions. For example, the solutions of an instance of the Satisfiability Problem (SAT) are the models of the given propositional formula. For combinatorial optimisation problems, the solution set can be defined in various ways, for example, it may comprise only candidate solutions with optimal solution quality, or candidate solutions that satisfy certain logical conditions or even all candidate solutions. Note that SLS algorithms for optimisation problems do not necessarily terminate as soon as a solution is found, but they return only candidate solutions that are guaranteed to be an element of the given solution set. Solutions as defined here are often also referred to as *valid* or *feasible candidate solutions*.

**Solution Density:** Relative frequency of solutions within the search space of a given problem instance; the solution density of a search space of size $\#S$ with $k$ solutions is defined as $k/\#S$. Problem instances with higher solution density tend to be easier to solve for stochastic local search algorithms.

**Solution Probability:** Probability with which a Las Vegas algorithm finds a solution of a given problem instance within a given time; also often referred to as *success probability*.

**Solution Quality:** Value of the objective function for a given optimisation problem, typically as applied to a candidate solution. The *relative solution quality* of a candidate solution with solution quality $q$ is defined as $q/q^* - 1$ for minimisation problems if $q^*$, the quality of an optimal (or quasi-optimal) solution, is not equal to zero, and as $(1 + q)/(1 + q^*)$ otherwise. Relative solution quality is useful in empirically evaluating the performance of optimisation algorithms across sets of instances with different optimal solution qualities; small relative solution quality values are often specified in percent, where a value of 1% is equivalent to a relative solution quality of 0.01. Relative solution quality is often simply referred to as solution quality.

**Solution Quality Distribution (SQD):** Probability distribution of the solution quality achieved by an optimisation Las Vegas algorithm within a fixed run-time. An *asymptotic SQD* is the solution quality distribution obtained in the limit as run-time approaches infinity; asymptotic SQDs are useful for the analysis of optimisation LVAs that are not probabilistically asymptotically complete, such as simple iterative improvement algorithms or constructive search methods.

**SQT (Solution Quality over Time):** Development of a statistical measure of the solution quality over the run-time of an optimisation Las Vegas algorithm; statistical measures commonly used in this context include the mean, median and quantiles of the solution quality distribution for the respective run-time. SQTs summarise the underlying bivariate run-time distributions and reflect the tradeoff between run-time and solution quality for the given algorithm on a specific problem instance. When reporting SQTs, care should be taken to reflect the variability of solution quality for each run-time, for example, by showing SQT curves for several quantiles; it is often advisable to additionally study solution quality distributions and qualified run-time distributions, which reflect different aspects of the underlying bivariate RTD.

**Stagnation:** Situation in which the *efficiency* of a search process decreases, where efficiency is formally defined based on the rate of increase over time in the solution probability of a given Las Vegas algorithm for a given problem instance. In the case of stochastic local search algorithms, stagnation occurs when the search process gets trapped (permanently or for a substantial amount of time) in a non-solution area of the given search space. Stagnation can typically be detected based on the run-time distributions of a given SLS algorithm; it can often be avoided or overcome by means of additional diversification mechanisms.

**Statistical Hypothesis Test:** Formal method for assessing the validity of statements about relations between or properties of sets of statistical data. For example, the *Mann-Whitney U-test* can be used to determine whether two stochastic local search algorithms applied to the same problem instance show statistically significant differences in median run-time. The statement to be tested (or its negation) is called the *null hypothesis* of the test. The *significance level* determines the maximum allowable probability of incorrectly rejecting the null hypothesis, while the probability of correct acceptance is bounded from below by the *power* of the test; for a given test, the power determines the required *sample size*. When applying statistical tests, often a significance level of 0.05 and a power of at least 0.8 is used. The application of a test to a given data set results in a *p-value*, which represents the probability that the null hypothesis is incorrectly rejected. Statistical hypothesis tests that are useful in the empirical analysis of the performance and behaviour of SLS algorithms include the *Mann-Whitney U-test*, the *binomial sign test*, the *Wilcoxon matched pairs signed-rank test*, *Spearman's rank order test*, the $\chi^2$- and *Kolmogorov-Smirnov goodness-of-fit tests* and the *Shapiro-Wilk normality test*.

**Steepest Ascent Mildest Descent (SAMD):** An early stochastic local search algorithm for the Maximum Satisfiability Problem that is conceptually closely related to Tabu Search. SAMD is based on a simple iterative best improvement algorithm, but unlike in Simple Tabu Search, only variables flipped in non-improving search steps are declared tabu.

**Stochastic Local Search (SLS) Algorithm:** Local search algorithm that makes use of randomised choices in generating or selecting candidate solutions for a given instance of a combinatorial problem. SLS algorithms may use random choices for search initialisation and/or the computation of search steps.

**Stochastic Search:** Search paradigm that makes use of randomised decisions. Stochastic local search algorithms as well as randomised systematic search algorithms fall into this general class of search methods.

**Success Probability:** Probability with which a Las Vegas algorithm finds a solution of a given problem instance within a given time; also often referred to as *solution probability*.

**Systematic Search:** Search paradigm under which the entire search space of a problem instance is traversed (or pruned) in a systematic manner, which renders the respective algorithms complete (unless the search process is terminated prematurely). Many systematic search algorithms perform some form of *tree search*, where each node $n_s$ in the search tree corresponds to a partial candidate solution $s$, and the children of $n_s$ represent all (feasible) partial candidate solutions that can be obtained by adding one solution component to $s$. When performed in a depth-first manner, this type of search can be seen as a constructive search process with backtracking.

**Tabu Search:** SLS method that strongly exploits memory of the search history for guiding the search process. The most basic form of tabu search, *Simple Tabu Search*, is obtained by enhancing an iterative improvement algorithm with a form of short-term memory that enables it to escape from local optima. In Simple Tabu Search, typically the solution components modified in a search step are declared *tabu* for a fixed number of subsequent search steps (*tabu tenure*); during their tabu tenure, solution components may not be added to or removed from the current candidate solution. An aspiration criterion can be used to override the tabu status of solution components. Simple tabu search algorithms are often enhanced by different types of intensification and diversification strategies. Tabu search algorithms are among the best-performing stochastic local search algorithms for many combinatorial problems.

**Tabu Tenure:** Important parameter of tabu search algorithms; the tabu tenure corresponds to the number of search steps for which the tabu status of a solution component $c$ is maintained after $c$ has been declared tabu and thereby determines the amount of time for which $c$ cannot be modified.

**Tardiness:** Amount of time by which a job in a scheduling problem is completed after its due date. In practical applications, tardiness often incurs additional costs, such as contractual penalties due to late deliveries.

**Termination Condition:** Component of a stochastic local search algorithm that determines when the search is ended; formally specified in the form of a predicate of the current candidate solution as well as partial historical or statistical information on the search trajectory. Typical termination conditions are satisfied if a solution of a decision problem is reached or a given solution quality of an optimisation problem has been achieved, if a certain bound on run-time or number of search steps has been exceeded, or if no improvement in evaluation function value has been observed for a certain amount of time.

**TMCH:** Stochastic local search algorithm for the Constraint Satisfaction Problem (CSP); a simple tabu search algorithm based on the Min-Conflicts Heuristic, in which after each search step, the respective variable/value pair is declared tabu for a fixed number of steps.

**Transition Action:** Action that is performed when a specific transition between two states of a Generalised Local Search Machine (GLSM) occurs. Transition actions may be used for modifying global parameters, such as the temperature parameter in simulated annealing algorithms, for communication between individual GLSMs in co-operative GLSM models or for input/output functionality.

**Transpose Neighbourhood:** Neighbourhood relation on permutations (or orderings) of solution components under which two permutations are direct neighbours if, and only if, one can be obtained from the other by interchanging two elements at adjacent positions. This neighbourhood relation is sometimes used in scheduling problems; however, local search algorithms based on the transpose neighbourhood perform often substantially worse than those based on the insertion or interchange neighbourhood.

**Travelling Salesman Problem (TSP):** Also known as *Travelling Salesperson Problem*; a prototypical combinatorial optimisation problem in which given an edge-weighted graph $G$, the objective is to find a cyclic path (also called *tour*) that visits every node in $G$ exactly once and whose total edge-weight is minimal. Often, the TSP is restricted to the special case where the vertices in $G$ (which are also called *cities*) correspond to points in the Euclidean plane, $G$ is fully connected, and the weight of any edge is defined as the Euclidean distance between the two points that correspond to its incident vertices (*Euclidean TSP*). The TSP (in general and in the Euclidean case) is $\mathcal{NP}$-hard. Stochastic local search algorithms are amongst the best known methods for solving hard and large TSP instances.

**TS-GH:** State-of-the-art tabu search algorithm for the finite discrete Constraint Satisfaction and Maximum Constraint Satisfaction Problems; TS-GH is based on an iterative best improvement procedure that in each step changes the value assigned to a variable appearing in a currently unsatisfied constraint in order to minimise the number (or total weight) of unsatisfied constraints.

**TS-NS-JSP:** State-of-the-art tabu search algorithm for the Job Shop Scheduling Problem. TS-NS-JSP uses a strongly reduced neighbourhood relation; conceptually, this algorithm is quite similar to TS-NS-PFSP.

**TS-NS-PFSP:** State-of-the-art tabu search algorithm for the Permutation Flow Shop Scheduling Problem (PFSP), which strongly exploits neighbourhood pruning for the PFSP and occasionally restarts the local search process at the incumbent candidate solution.

**TSPLIB:** A library of widely used benchmark instances for the Travelling Salesman Problem (TSP) and several related problems. As of September 2003, the largest TSP benchmark instance from TSPLIB solved to optimality has 15 112 vertices.

**Uniform Random *k*-SAT:** Prominent class of benchmark instances for the Satisfiability Problem (SAT); every clause in a Uniform Random $k$-SAT instance consists of exactly $k$ literals that are chosen uniformly at random from a given set of $n$ propositional variables and their negations, where no clause may contain multiple copies of the same literal or a variable and its negation. The mean empirical hardness of Uniform Random $k$-SAT instances strongly depends on the ratio between the number of clauses and variables.

**Uninformed Random Picking:** Simple stochastic search method that repeatedly samples the given search space by selecting a candidate solution uniformly at random. Many stochastic local search algorithms use Uninformed Random Picking for search initialisation.

**Uninformed Random Walk:** Simple SLS method that in each search step selects a candidate solution $s'$ uniformly at random from the neighbourhood of the current candidate solution $s$. Because it lacks heuristic guidance, Uninformed Random Walk alone performs very poorly, but this simple SLS method and its variants are not only of considerable theoretical interest, but are also sometimes used as diversification mechanisms in high-performance stochastic local search algorithms.

**Unit Propagation:** Polynomial preprocessing method for the Satisfiability Problem (SAT); eliminates *unit clauses*, that is, clauses that contain only one literal, from a given CNF formula and propagates the effects to all other clauses containing the same variable. Unit propagation plays a crucial role in high-performance systematic search algorithms for SAT.

**Valid Candidate Solution:** Alternative term for solution, that is, for a candidate solution that is an element of the solution set of a given stochastic local search algorithm and problem instance.

**Variable Depth Search (VDS):** Iterative improvement method that is based on *complex search steps*, which are obtained by concatenating a variable number of *simple search steps*. VDS forms the basis of many of the best-performing iterative improvement algorithms, such as the well-known Lin-Kernighan Algorithm for the Travelling Salesman Problem.

**Variation Coefficient:** Statistical measure of the variation of a given probability distribution. The variation coefficient $vc$ is defined as the ratio of the standard deviation and the mean of the given distribution. Compared to the standard deviation, the variation coefficient $vc$ has the advantage that it is invariant w.r.t. multiplicative scaling of the underlying distribution.

**Variable Neighbourhood Search (VNS):** Stochastic local search method based on the general idea of changing the neighbourhood relation during search. Examples of VNS algorithms include *Variable Neighbourhood Descent (VND)*, an iterative improvement method that systematically switches between different neighbourhood relations, and *Basic VNS*, which can be seen as an iterated local search method that systematically modifies the strength of its perturbation procedure.

**Walk Probability:** Important parameter of randomised iterative improvement (RII) algorithms; the walk probability determines the probability with which at any given step of an RII algorithm a random walk step is performed. This parameter also appears in various high-performance hybrid stochastic local search algorithms for the Satisfiability Problem and the finite discrete Constraint Satisfaction Problem.

**WalkSAT:** Prominent family of high-performance stochastic local search algorithms for the Satisfiability Problem (for CNF formulae). WalkSAT is based on the 1-flip neighbourhood; in each search step, first a currently unsatisfied clause $c$ is selected uniformly at random, and then a heuristically chosen variable from $c$ is flipped, rendering $c$ satisfied. WalkSAT algorithms differ in the selection mechanisms used for choosing the variable from $c$ to be flipped. The term is also used to refer to one specific WalkSAT algorithm, *WalkSAT/SKC*, which is based on a randomised best improvement variable selection mechanism.

**WMCH:** Well-known stochastic local search algorithm for the finite discrete Constraint Satisfaction Problem; a variant of the Min-Conflicts Heuristic (MCH) that can be seen as a randomised iterative improvement method, which in each search step first selects a variable that appears in a currently unsatisfied constraint, and then with a certain probability sets this variable to a randomly chosen value, and otherwise chooses a value that minimises the number of violated constraints (as in MCH).