Currently the most popular and effective approaches to fully automated planning are:

Boolean satisfiability

Boolean satisfiability (SAT)-the problem of determining whether there exists an assignment satisfying a given Boolean formula-is a fundamental intractable problem in computer science. SAT has many applications in electronic design automation (EDA), notably in synthesis and verification. Consequently, SAT has received much attention from the EDA community, who developed algorithms that have had a significant impact on the performance of SAT solvers. EDA researchers introduced techniques such as conflict-driven clause learning, novel branching heuristics, and efficient unit propagation. These techniques form the basis of all modern SAT solvers. Using these ideas, contemporary SAT solvers can often handle practical instances with millions of variables and constraints. The continuing advances of SAT solvers are the driving force of modern model checking tools, which are used to check the correctness of hardware designs. Contemporary automated verification techniques such as bounded model checking, proof-based abstraction, interpolation-based model checking, and IC3 have in common that they are all based on SAT solvers and their extensions.

Forward state-space search with carefully crafted heuristics

One of the simplest planning strategies is to treat the planning problem as a path planning problem in the **state-space graph**.
A **forward planner** with a heuristic function is a way to inform searches about directions from the initial state to state that satisfies a goal description.
A* search is an informed (heuristic based) search algorithm used for path-finding and graph traversal. It combines the advantages of both Dijkstra's algorithm (in that it can find a shortest path) and Greedy Best-First-Search (in that it can use a heuristic to guide search). It combines the information that Dijkstra's algorithm uses (favoring vertices that are close to the starting point) and information that Best-First-Search uses (favoring vertices that are closer to the goal). It is optimal and the efficiency is dependent on quality of heuristic. Peter Hart, Nils Nilsson and Bertram Raphael of Stanford Research Institute (now SRI International) first described the algorithm in 1968.

Search using a planning graph

GraphPlan (Blum & Furst, 1995) was the first planner to use planning graph techniques. Before GraphPlan came out, most planning researchers were working on partial order planners (plan space planners *) .like POP, SNLP, UCPOP, etc. GraphPlan caused a sensation because it was so much faster. Many subsequent planning systems have used ideas from it either directly as close decendants of GraphPlan or by using the Planning Graph representation in some guise to improve the encoding of the planning problem most notably for SAT-based planning. GraphPlan's place in history is secured by its critical role in the development of reachability heuristics* for heuristic search planners by approximating the search tree rooted at a given state. Heuristic search planners have dominated the "satisficing planner" track of IPC planning competitions for the last 8 years.

References

1. http://ieeexplore.ieee.org/document/7225110/
2. http://www.cs.toronto.edu/~sheila/2542/f10/slides/CSC2542f10_plangraphs_4pp.pdf
3. http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
4. https://en.wikipedia.org/wiki/A*_search_algorithm