Research Review:

AI planning was culmination of investigations into state-space search, theorem proving, and control theory [1]. STRIPS was the first major planning system developed at SRI. This review looks at the some of the key findings and concepts of STRIPS and further explores the topic of automatic program synthesis.

STRIPS (STanford Research Institute Problem Solver) is a problem solver that attempts to find a series of operators in a world model state space to transform a given initial world model into a model that proves the given goal formula [Fikes, Nilsson, 1971]. STRIPS uses resolution theorem prover to answer questions about particular models and uses means-ends analysis to reach desired goal-satisfying model [Fikes, Nilsson, 1971]. In STRIPS, a model us represented by a set of well-formed formulas of the first-order predicate calculus. Solutions are built from basic elements called operators, and each operator corresponds to an action routine [Fikes, Nilsson, 1971]. There have been some implementations of problem-solvers that exclusively use theorem-proving methods to look for sequence of operators, but they struggle with "frame problem" [Fikes, Nilsson, 1971]. STRIPS overcomes this problem by separating processes of theorem proving from those of searching through state space. Searching through space is done by GPS-like means-end analysis strategy [Ernst, Newell, 1969]. GPS strategy extracts "differences" between the present world model and the goal and identifies operators that relevant to reducing these differences [Ernst, Newell, 1969].

STRIPS uses heuristic evaluation functions that consider factors such as number of remaining goals on the goal list, the number and types of predicates in the remaining goal formulas, and the complexities of the differences at nodes to decide which successor nodes to select next [Fikes, Nilsson, 1971].

One of the topics of future research paper touches on is the concept of synthesizing more complex procedures than just simple linear sequences of operators using iteration and conditional branching. STIRPS should be able to generate computer programs, and this comes under the domain of automatic program synthesis. A program synthesizer is system that takes a relation between input and output and tries to produce a program that is guaranteed to satisfy the relationship, in-turn eliminating the need for debugging or verification [Manna, Waldinger, 1970]. At the heart of automatic program synthesis is theorem-proving. In order to construct a program satisfying certain specifications, a theorem induced by those specifications is proved, and the desired program is extracted from the proof [Manna, Waldinger, 1970]. The automatic program synthesis generally is given an input predicate and an output predicate. These predicates provide the specifications for the program to be written [Manna, Waldinger, 1970]. Induction principle is the most commonly used to prove theorems about natural numbers, and analogues can also be used to prove other data structures such as lists, trees, and strings [Manna, Waldinger, 1970]. There are multiple induction principle methods and the choice determines which program will be constructed.

Another popular program for automatic program writing is called “PROW.” PROW accepts the specifications of the program in the language of predicate calculus, decides which algorithm to use and then produces a LISP program [Waldinger, Lee, 1969]. PRUW can be used to write any programs by modifying the part that translates an algorithm into a program. PROW stores instructions to program as axioms. The user gives a relationship between the input and output variables as a well-defined formula in the first order predicate calculus and PROW comes up with a theorem to prove the relationship. Once theorem prover outputs a “proof” of the relationship, another program processes the proof and gives the desired program [Waldinger, Lee, 1969].

The research into automatic program writing domain is on-going and provides excellent opportunity to improve the programming experience in-general whereby automated program writers could ease the burden of knowing a particular computer language and helping in debugging of algorithm implantations [Waldinger, Lee, 1969].

References:

1. "Artificial Intelligence: A Modern Approach" 3rd edition Chapter 10, pg. 393.
2. Fikes, R. E. and Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. AIJ, 2(3–4), 189–208.
3. Manna, Z. and Waldinger, R. Towards automatic program synthesis. Comm. ACM. 14, No. 3 (March 1971).
4. Waldinger, R. and Lee, R. PROW: A step toward automatic program writing. Proc. lnt'l. Conf. Artificial Intelligence, Washington, D.C. (May 1969).
5. Ernst, G. and Newell, A. GPS: A Case Study in Generality and Problem Solving. ACM Monograph Series. Academic Press, New York, New York, 1969.