Automated Process Planning for CNC Machining

Christian Fritz

■ *This article describes an application of AI planning to the problem of automated process planning for machining parts, given raw stock and a CAD file describing the desired part geometry. Researchers at PARC have found that existing planners from the AI community fall short on several requirements, most importantly regarding the expressivity of state and action representations and the ability to exploit domain-specific knowledge to prune the search space. In this article I describe the requirements for this application and what kind of results from the planning community helped most. Overall, in this project as well as others, I found that even significant results from domain-independent planning may not be relevant in practice.*

"Unsurprisingly, AI planning in industry is domain-specific."

A large portion of today's industrial manufacturing relies on subtractive machining, a process in which a fast-spinning tool successively removes material from raw stock, for example, a block of aluminum, in order to arrive at a part geometry as specified by a design engineer in a computer-aided design (CAD) file. Planning for this process typically involves identifying the sequence of orientations in which the work piece needs to be fixed, identifying the sequence of tools to be used in each orientation, identifying the part of the volume to be removed in each step of each orientation, and identifying the machine to use for each step of each orientation. Until now, this planning process was not automated but done by humans.

The Defense Advanced Research Projects Agency (DARPA) recognized that this lack of automation was a source of delays in the design and production of new vehicles and requested proposals to address this issue and to provide automated manufacturing feedback back to designers. At Palo Alto Research Center (PARC), researchers recognized the potential business value to designers as well as manufacturers, and this value proposition was validated during project execution by presenting early prototypes of the software to potential users. The objective of PARC's uFab project hence was to create a software tool that, given just a CAD file and a representation of available machines and tools, generates a process plan in real time. While work in this area had been done in the 1980s under the name computer-aided process planning (CAPP) (Alting and Zhang 1989), none of the approaches that were pursued then resulted in a fully automated solution. A major shortcoming of these systems was their reliance on features, recognizable configurations of faces on a part such as pockets, slots, and holes, in order to represent states and actions. In these approaches, planning amounted merely to selecting from a set of predefined operations to make each of the recognized features. This reliance on feature-based representations hindered their broad applicability to parts that could not be easily described as a combination of features or where feature recognition was difficult, ambiguous, or error prone.

Feature-Free Process Planning

My group therefore went with an entirely different representation. Enabled by much more computational power and larger memory than in the 1980s, the approach uses a hybrid representation of faces and voxels — three-dimensional pixels. The CAD geometry to be analyzed is discretized into a large number of voxels (on the order of 200 per coordinate axis). Given this representation, actions are characterized by a tool and an orientation, and their effect is described by the removal volumes, that is, the sets of voxels that can be reached by the tool in the given orientation, given its length and diameter, without colliding with the desired geometry. Computing these removal volumes requires geometric reasoning, and is much slower than one would want in an inner loop of a planner (ca. 200 ms). For this reason we precompute all maximal removal volumes ahead of planning, and then only intersect and union these volumes during planning.

Taken together, these properties of the representation rendered Planning Domain Definition Language (PDDL)–based planners unusable for purposes or this research. These planners do not have the representational expressivity to capture geometric operations, nor are they well suited to computing large intersections (millions of logical conjunctions) and unions (millions of logical disjunctions). Nor did they need to. Given that my colleagues and I worked on this problem for multiple years, which is not uncommon in industry, it was entirely acceptable to design and implement a domain-specific planner that uses fast C-implementations for required set operations, the z-buffer algorithm (Rossignac and Requicha 1986) for geometric reasoning, and intelligent precomputation of action effects where possible. Relative to these domain-specific needs, implementing the actual search used for planning was the easy part. My colleagues and I used a combination of beam and weighted A* search, and exploited some insights from solutions for the set-cover problem. Since the value-proposition was based on an interactive experience, the challenge was to balance optimality with response time for the user, but in addition to the usual search parameters like beam width and heuristic weight, my colleagues and I had broad domain-specific knowledge at our disposal, which proved most powerful.

As a result, PARC is now able to offer a web-based service for automated process planning that can save machine shop owners several hours of manual work each day, resulting in clear monetary value. This benefit is most pronounced for machine shops that specialize on "high-mix, low-volume" business, that is, where only a relatively low quantity of each newly quoted part is cut.

Lessons Learned

Catalyzed by the international planning competition, the AI planning community has accomplished great speedups in domain-independent planning for a number of planning categories. However, in my experience with this industrial application as well as others, domain-independence rarely matters in practical applications. In industry, almost by definition, people have industry-specific, that is, domain-specific, planning needs. If the planning community would like to help industry with these needs, then a shift in focus may be advisable. The type of research that I believe industry would most benefit from involves more expressive representations, including metalanguages for creating domain-specific languages to use for state and action space representation, hybrid approaches between declarative planning and procedural programming, such as TLPlan (Bacchus and Kabanza 1998), Golog (Levesque et al. 1997), or even procedural attachments, as well as fundamental insights about combinatorial search that are independent of the representation. Any advances that are specific to domain-independent planning in PDDL, such as the powerful deletion heuristic, are unlikely to move the scale in practice, at least for me. The competition is simply too high: dealing with the NP or PSPACE hardness of planning depends on the ability to exploit structure, and knowing the domain simply puts domain-specific planners in an (exponentially) unfair advantage over domain-independent approaches. As a result, the business model for domain-independent planners is questionable, and I believe aspiring graduate students are ill advised to focus too much on them if they care about the practical long-term impact of their work.

Acknowledgements

The work on this article was performed while the author was at Palo Alto Research Center, Palo Alto, CA, USA.

References

Alting, L., and Zhang, H.-C. 1989. Computer Aided Process Planning: The State-of-the-Art Survey. *International Journal of Production Research.* 04/1989; 27(4): 553–585. dx.doi.org/10.1080/00207548908942569

Bacchus, F., and Kabanza, F. 1998. Planning for Temporally Extended Goals. *Annals of Mathematics and Artificial Intelligence* 22(1–2):5–27, 1998. dx.doi.org/10.1023/A:1018985923441

Levesque, H. J.; Reiter, R.; Lesperance, Y.; Lin, F.; and Scherl, R. B. 1997. GOLOG: A Logic Programming Language for Dynamic Domains. *Journal of Logic Programming* 31(1–3): 59–83. dx.doi.org/10.1016/S0743-1066(96)00121-5

Rossignac, J. R., and Requicha, A. A. G. 1986. Depth-Buffering Display Techniques for Constructive Solid Geometry. *IEEE Computer Graphics and Applications* 6(9) (September): 29–39. dx.doi.org/10.1109/MCG.1986.276544

Christian Fritz until recently led the Representation and Planning area at PARC. Prior to assuming that role he was a research scientist, working on various applied planning projects. Prior to PARC he was a postdoctoral researcher at USC ISI. Fritz earned his Ph.D. in computer science from the Knowledge Representation group at the University of Toronto, Canada, and his MS and BS degrees in computer science from RWTH Aachen, Germany.