Improving a Plan Library for Real-time Systems Using Nearly Orthogonal Latin Hypercube Sampling

Robert Holder
University of Maryland Baltimore County
holder1@umbc.edu

Abstract

Computing solutions to intractable planning problems is particularly problematic within real-time domains. One approach to this challenge includes off-line computation, such as precomputing a plan library. However, because complex domains preclude creating a comprehensive library, a system must choose a subset of all possible plans to include in the library. Strategic selections will reduce the probability that a system encounters a situation for which it does not have an appropriate plan in the library to either apply directly or adapt.

Choosing variable values using *Latin hypercubes* is a technique used to reduce the number of test cases required in order to validate complex systems. Here we discuss the application of a variation of this technique, *nearly orthogonal Latin hypercubes*, to planning spaces in order reduce the number of plans a system must cache in its library.

Introduction

One approach to facilitating real-time planning within dynamic environments is to precompute plans for environments that the system is likely to encounter. The system expects that a selected plan will either be "good enough" as is, or easily adapted to a plan that satisfies the requirements of the environment. To have a high probability of selecting a plan that is "good enough," the plan library must contain plans that represent solutions to a large variety of environments. Also, the library should avoid large "gaps" so that plan adaptation is feasible within real-time. Thus, the challenge in developing such a plan library is to select an appropriate sampling scheme, as well to determine the parameters to use as the dimensions of the parameter space.

Below we describe our approach to finding an appropriate sampling scheme to generate a plan library for a large-scale, real-time distributed planning problem.

Sampling schemes

Figure 1 displays a two-dimensional example of several schemes, where each dimension represents the value space of one parameter. However, each of these schemes generalizes to an arbitrary number of parameters.

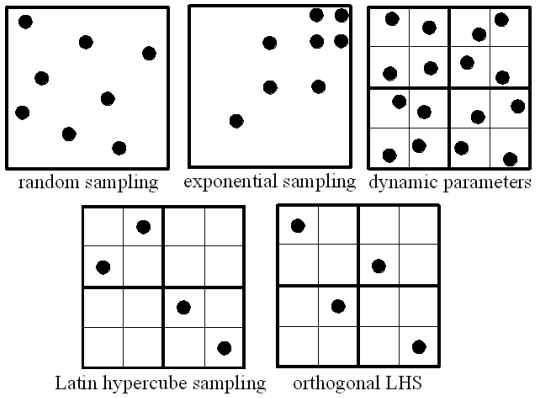

Figure 1: Sampling schemes for two parameters. Lines represent major and minor divisions of the space as defined by each scheme. Each point represents a vector of the two parameter values.

We develop a plan as a function of a *vector* consisting of a set of selected parameter values. In order to compute a plan library, we generate a set of vectors. Thus, the manner in which we sample the parameter space determines the distribution of plans in the library.

Latin hypercube sampling

In Latin hypercube sampling (McKay, Beckman, & Conover 2000), every region of each parameter is guaranteed to be uniformly represented. Typically, each parameter's value range is divided into regions, and a sample is taken from each region. These vectors within the parameter space are generated by randomly choosing samples from each parameter. In figure 1, each parameter is divided into four regions, and one possible distribution of selected vectors is shown. This scheme is not computationally intensive, but large portions of the parameter space are unrepresented.

Dynamic parameters

Kwok *et al.* describe dynamic parameters as inputs into a scheduling problem for which the output is a mapping of applications onto computer processors (Kwok *et al.* 2006). Their technique begins with stratified sampling (McKay, Beckman, & Conover 2000), in which the multi-dimensional

parameter space is divided into regions, and several representative vectors are randomly chosen from each region. A solution is generated for each vector, and then each solution within a region is evaluated for all the vectors in the region. Each region is then represented by the best solution generated from the samples within the region.

Exponential sampling

Our previous research (Holder *et al.* 2006) takes inspiration from Kwok *et al.*'s dynamic parameter approach. However, we only use one region, and, instead of randomly selecting representative vectors, we choose an exponential spacing distribution of the vectors within the region. Our motivation is to create a plan library containing a concentration of plans in regions of the parameters space where plan adaptation is difficult.

Orthogonal Latin hypercube sampling

Orthogonal Latin hypercube sampling (OLHS) (Ye, Li, & Sudjianto 1998) builds upon LHS by preventing large gaps within the parameter space. As in LHS, each parameter is partitioned into regions and a vector is created by combining samples from each parameter. However, the parameter space is divided into subregions that must also be sampled uniformly. Thus, we get a distribution such as that in figure 1 where each region of a parameter is sampled once, and each quadrant of the region is also only sampled once. Although this technique does offer a uniform distribution with few samples, it is computationally intensive to apply to parameter spaces with many dimensions.

Nearly orthogonal Latin hypercube sampling

Nearly orthogonal Latin hypercube sampling (Cioppa 2002) attempts to offer many of the benefits of OLHS with less computational cost. Cioppa shows that as the dimensionality of OLHS increases, the resulting distributions do start to exhibit gaps in the parameter space, or, as Cioppa describes it, bad "space-filling." In order to address both issues, nearly orthogonal LHS relaxes the orthogonality constraint. This scheme allows for extensions to higher dimensional parameter spaces, better space-filling, and faster computation.

Application to large-scale real-time planning

We consider an instance of the FAME domain (Dale & desJardins 2007) consisting of a map with 49 regions, each of which contains approximately 600 broadcast stations that require monitoring at various times. Within each region there are 50 mobile sensors that can be deployed to serve monitoring requests. Our system will have to adapt in real-time to various factors such as incoming monitoring requests and sensor failures. The system will be evaluated by the percentage of all requests and high-priority requests that it satisfies.

Plan distribution

A good distribution of precompiled plans will enhance the ability for our sensors to service requests within this dynamic environment. To test this hypothesis, we will populate a plan library using various sampling schemes.

We expect nearly orthogonal LHS sampling will generate a well-distributed plan library. However, in previous experiments (Holder *et al.* 2006), we found that plan adaptation was much more difficult within specific regions of the parameter space. It is likely the optimal scheme will entail defining and applying nearly orthogonal LHS sampling to subregions of the parameter space.

Defining relevant plan parameters

The obvious dimensions of the parameter space include environment variables such sensor location, broadcast station location, and monitoring appointment time. We may also consider stochastic measures such as expected sensor failure rate. For example, the system may receive information that a specific type of sensor has become unreliable. Given this, a system might construct its plan library to increase contingencies when generating vectors for high values in the sensor failure rate dimension. We may also consider adding planning algorithm parameters as a dimension of our parameter space.

Conclusion

Our research goal is to increase the efficiency of a large-scale, real-time distributed planning problem by developing a useful precomputed plan library. We expect that nearly orthogonal LHS sampling will result in an effective tradeoff between the size of the plan library and the quality of plans available for situations that our system will encounter. We have briefly described some sampling schemes and a domain in which we will apply them.

References

Cioppa, T. M. 2002. *Efficient Nearly Orthogonal And Space-Filling Experimental Designs For High-Dimensional Complex Models*. Ph.D. Dissertation, Naval Postgraduate School.

Dale, M., and desJardins, M. 2007. The FAME problem domain for distributed planning. In *Working Notes of the AAAI Fall Symposium on Regarding the Intelligence in Distributed Intelligent Systems*.

Holder, R.; Pascale, C.; Dale, M.; Daley, R.; Chong, E.; Shestak, V.; Siegel, H.; and Marinescu, D. 2006. Company Resource Management (CRM) Algorithm Description Document. ARMS Final Report.

Kwok, Y.-K.; Maciejewski, A. A.; Siegel, H. J.; Ahmad, I.; and Ghafoor, A. 2006. A semi-static approach to mapping dynamic iterative tasks onto heterogeneous computing systems. *Journal of Parallel and Distributed Computing* 66:77–98.

McKay, M. D.; Beckman, R. J.; and Conover, W. J. 2000. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 42(1):55–61.

Ye, K. Q.; Li, W.; and Sudjianto, A. 1998. Algorithmic construction of optimal symetric Latin hypercube designs. *Journal of Statistical Planning and Inference* 90(1):145–159.