XPlan: Experiment Planning for Synthetic Biology

Ugur Kuter† and **Robert P. Goldman**† and **Daniel Bryce**† and **Jacob Beal**‡
Matthew DeHaven† and Christopher S. Geib† and Alexander F. Plotnick† and Tramy Nguyen‡ and Nicholas Roehner‡
†SIFT, LLC
Minneapolis, MN, USA
{dbryce, rpgoldman, ukuter, mdehaven, aplotnick, cgeib}@sift.net
‡Raytheon BBN Technologies
Cambridge, MA, USA
jacob.beal@ieee.org, nicholas.roehner@raytheon.com, tramy.nguy@gmail.com *

Abstract

We describe preliminary work on XPlan, a system for experiment planning in synthetic biology. In synthetic biology, as in other emerging fields, scientific exploration and engineering design must be interleaved, because of uncertainty about the underlying mechanisms. Through its experiment planning, XPlan provides a coordinating linchpin in DARPA's Synergistic Discovery and Design (SD2) platform to automate scientific discovery, closing the loop between multiple machine learning analysis and biological design tools and wet labs to guide the discovery and design process. To accomplish this, XPlan combines design of experiments techniques with hierarchical planning, based on the Shop2 planner, to develop experimental plans that are directly executable in highly automated wet labs and to project experimental costs. In particular, XPlan formulates experimental designs and translates them into goals representing biological samples, then uses Shop2 to plan construction and measurement of samples using available laboratory resources. In ongoing work, we are developing probability models that will support value of information computations to optimize experimental plans.

1 Introduction

In the field of synthetic biology, as with other emerging fields of engineering, scientific exploration and engineering design are intimately entwined. Unlike established fields of engineering, synthetic biology has only highly uncertain and incomplete mechanistic models. As a result, engineering synthetic biological systems is an incremental process in which the production of designs is closely interleaved with execution of experiments to assess the success of those designs and data analysis to identify factors and mechanisms responsible for design successes and failures.

Organization and planning of synthetic biology experiments is currently done almost entirely by hand. Several ongoing developments, however, are rapidly increasing the need for automation assistance in experiment planning. More and more laboratory automation is becoming available, increasing the scale and complexity of experiments that can be performed. Automation and information technology are supporting new business models with laboratory work done by technicians or outsourced to a "lab for hire." Finally, new "multiplexing" protocols allow many tests to be conducted on a single experimental sample, and multiple experimental samples to be processed in parallel. In all of these cases, the growth in scale and complexity are rapidly outstripping the abilities of humans to create detailed experimental plans and to hand-curate the relationships between those plans and the large collections of data they generate. Furthermore, experiments are still costly both in money and time, and the space a researcher wishes to explore is often much larger than the number of samples that can be tested, so there are opportunities for automation to assist in optimizing the value of information from each sample, potentially even dynamically based on partial results from an experiment in progress.

This paper explains how our XPlan planner, based on Hierarchical Task Network (HTN) planning, addresses these issues by providing automation support for experiment planning. In the next section, we describe the class of discovery and design problems addressed by XPlan, and the challenges they pose. We then explain how our HTN approach, based on the Shop2 planner, addresses these challenges, and describe our early-stage work on optimizing the expected value of information while planning experiments. Finally, we summarize and describe some next steps.

2 Synergistic Discovery and Design (SD2)

Synthetic biology is the systematic engineering of living organisms to perform desired functions. For example, biological sensors have applications in sensing biological, chemical, and radioactive weapons, and pathogens; effectors have applications in chemical synthesis and cleanup, and targeted medical therapies. Because existing models for genetic structures, assembly, and expression are still relatively weak, however, synthetic biology necessarily involves both design and experimentation to assess the success of designs and identify factors responsible for success and failure.

DARPA's Synergistic Discovery and Design (SD2) program seeks to speed scientific and design processes through automated support for experiment planning, automated execution of experimental protocols across laboratories, and

*This work was supported by the Air Force Research Laboratory (AFRL) and DARPA under contract FA875017CO184. This document does not contain technology or technical data controlled under either U.S. International Traffic in Arms Regulation or U.S. Export Administration Regulations. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the AFRL and DARPA.

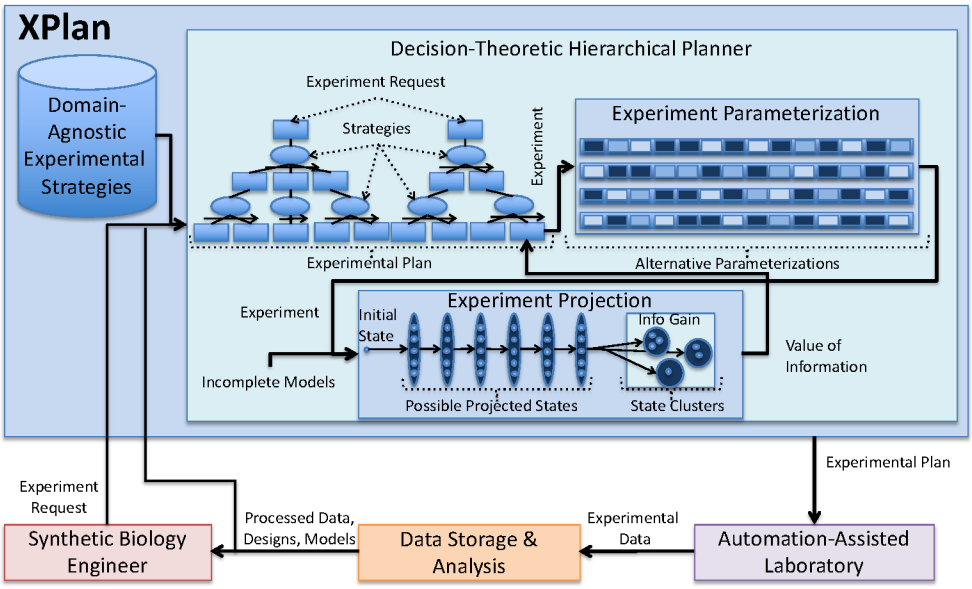

Figure 1: XPlan combines domain-agnostic strategies and domain-specific knowledge to expand experiment requests into executable plans, which are then parameterized and projected for VOI analysis. Plans are given to laboratories to execute, producing data that results in updated models and designs and new experiment requests.

high-speed, large-scale exploratory data analysis. Figure 1 shows the high-level architecture of XPlan, our hierarchical experiment planning system, which is a key part of the overall SD2 project. XPlan uses HTN planning to generate experimental protocols from synthetic biologists' expressions of experimental intent. It also translates the protocols it generates into executable forms so that they can be performed at different laboratories, which have different equipment, levels of automation, and processes.

XPlan also helps with data analysis, by storing information about the protocol in SynBioHub (McLaughlin et al. 2018; Madsen et al. 2016; Roehner et al. 2016), a standard synthetic biology semantic database. This enables labs performing protocols to accurately and consistently label the resulting data, immensely simplifying the process of data analysis. It also enables the operation of an automated pipeline for preliminary data formatting, labeling, and processing.

Experiment Planning for synthetic biology is challenging for a number of reasons:

- **Complex Systems and Incomplete Models:** The causal processes underlying biological mechanisms and their response to external stimuli are complex and only partially known. Designing biological circuits is difficult because of constraints such as the interactions of elements of the design with one another, and with existing biological functions (e.g., the design takes resources the cell needs to live). Therefore, decisions such as how long to incubate samples (to allow cells to multiply and respond to their environment) are difficult to make. Such decisions are made based upon the expertise of biologists without a formal causal model that would support the simulation/projection necessary for first-principles planning.
- **Replication:** Replicating experiments at multiple locations is critical in emerging fields. This follows from the incomplete knowledge described above: since we are not certain what environmental factors are most critical, and must be most carefully controlled, replication can provide even more information than in well-understood fields.
- **Large Sample Sizes:** Recent advances in biotechnology have reduced experiment costs through increased parallelization. In many cases, the cost of culturing hundreds of samples is only marginally more than the cost of a single sample. However, some operations such as sequencing the genome of a sample are not parallelizable. This is a challenge for planners because they must reason about many samples (objects). In some cases operations apply to all samples and do not impact the plan search branching factor. In other cases, the planner must select between subsets of samples for an operation, and hence cope with a large branching factor.

The problem of planning experiments for synthetic biology has several interesting features from a planning perspective:

- **Long Planning Windows:** Experiments often execute over the course of multiple days, providing a long window between planning episodes. Each planning episode involves selecting a new set of samples and conditions to investigate based upon the outcome of previous experiments. With several days to plan, it is possible to consider many possible plans. This allows for a more costly analysis of alternatives and shifts the emphasis from finding satisficing plans to finding high quality plans.
- **Multiple Levels of Abstraction:** Plan executors (laboratories) offer a range of robot and human executed primi-

tive actions. Each laboratory offers a layer of abstraction over the actions that go into experimental protocols, and those layers of abstraction vary between labs based on the extent of automation, equipment available, and management structure. This provides an interesting relationship between choice of performer and nature of the procedure. Also, by providing a common view on these differing institutions, XPlan can give real value to its users.

- **Managing Costs and Benefits:** Synthetic biology experiments are expensive, and because different labs' cost models are different, it can be difficult for biologists to predict the costs of performing a particular protocol at a particular facility or set of facilities. XPlan incorporates multiple labs' cost models into its HTNs to compute costs along with protocols. To help with the benefit side of the analysis, we are beginning to add *value of information* computation and guidance to XPlan to help biologists get the most useful information with their limited resources (both monetary and human).

3 Experiment Planning

Experiment planning involves two sub-problems: (1) experimental design, to select the samples and conditions to test, and (2) plan synthesis, to create the procedures that will construct and then measure the samples. In this paper, we focus on plan synthesis and describe our hierarchical planning approach based on our Shop2 system (Goldman and Kuter 2018; Nau et al. 2003; 2005).

Shop2's hierarchical planning approach is particularly well-suited to planning synthetic biology experiments. HTNs enable us to easily capture expert knowledge from biology researchers and formalize that knowledge for use in planning. A specific advantage of Shop2 is that it is a forward state progression planner: it performs task decompositions in the order those tasks will be executed in the world, while progressing the current state. Because it does forward state space planning, Shop2 always has a full model of the current state of the world (and the history that led to it). This full world state enables Shop2 to incorporate considerably greater expressive power – for example, capabilities for calling attached procedures, making axiomatic inferences, and performing numeric computations – than other HTN planners (e.g., UMCP (Erol, Hendler, and Nau 1994) and Sipe (Wilkins 1988)), that work with partial world models. Shop2 was developed this way for work on designing for manufacturability, which involved using CAD tools in projective planning. Such tools require full state descriptions, and typically are incapable of regression. For example, one cannot look at a machined part and reason to what was used to build it, but it's straightforward to give a blank and a design to a CAD tool to identify the required cuts and project the resulting part. Progression search also *potentially* allows for easier incorporation of informed heuristics, though at the moment the heuristics in XPlan are incorporated in the HTN preconditions. It is an open research question how to combine informed heuristics with such expressive preconditions, and with the task-based, as opposed to goal-based, semantics of Shop2 plans.[1]

[1]See Goldman (2009), for a discussion of the semantics of Shop2 task networks.

XPlan's plan library is divided into three components, broadly speaking. First is a high-level library of experimental strategies that is not specific to synthetic biology or to particular laboratories. These strategies aim to distribute experiments across laboratories for execution while minimizing variation, validating hypotheses, and determining parameters for designs during planning. Second is an abstract set of protocol components that are specific to synthetic biology, but not to particular lab configurations. Finally, there are methods that are specific to particular labs, and that enable our procedures to be translated into executable form. For example, some of these library components enable an XPlan-generated experimental protocol to be translated to Autoprotocol. Autoprotocol, developed by Transcriptic[2], is an executable JSON schema providing a domain specific language for automated wet lab operation.

Consider an experiment for measuring growth rate of a set of modified yeast strains over time via optical density (OD), which characterizes the amount of cells interfering with light shining through a sample. Experiment plans must first select the combination of biological factors that will be most informative. Examples for yeast strains include the modified genes, the yeast strain itself, small molecule concentrations, and other environmental factors (media type, temperature, humidity, etc.). These factors affect growth rate, which can be estimated across time by monitoring changing OD.

Listing 1 shows a Shop2 planning operator for the process of "provisioning" a replicate, i.e., collecting a sample from a particular strain of micro-organism (identified by `?resource`) in order to use it as a replicate in an experiment. A "replicate" is one of multiple copies of the same strain/conditions pair, used to ensure results are not lost due to mistake, and to provide sufficient data for later analysis. Intuitively, "provisioning" like selecting a cup of an ingredient for use in a recipe. The resource argument will be bound to a URI pointing to a SynBioHub entry representing a strain of yeast.

Listing 1: Example Shop2 sample provisioning operator definition.

```
(:op (!provision-replicate ?sid
                ?resource ?colony-type)
 :precond
   ((experiment-id ?ex-id)
    (assign ?sample-uri
      (make-sample-uri ?ex-id ?sid)))
 :add
   ((experiment-sample ?sid ?sample-uri)
    (derived-from ?sid ?resource)
    (resource ?colony-type ?sid)
    (sample-map
       (:source ?resource
          :destination ?sample-uri)))
  :cost 0.0)
```

This operator description highlights one of the expressive features of Shop2's HTNs not present in traditional planners: its ability to make external function calls during planning and incorporate the return values of those calls into its plan and

[2]https://www.transcriptic.com/

state. In the above example, the operator will call the function `make-sample-uri`, passing the values of two variables (the experiment id and the sample id) from the plan space as arguments and receive a newly-generated URI for the sample. Unlike classical or other HTN planners, the ability to make external function calls makes SHOP2 Turing-complete[3] and highly applicable to practical planning domains.

We also use SHOP2's facilities for computing action costs to compute experiment costs. SHOP2 allows an author to specify either static or dynamic cost functions in operator descriptions. The former is a fixed number across all instantiations of the operator description (e.g., the operator in Listing 1 has a static zero cost). The latter defines cost value as a function of the parameters from preconditions and task arguments. Listing 2 shows an operator whose cost is computed by looking up lab-specific costs for a growth method (`?meth`) and the lab's minimum sample size. This cost summary is then accumulated to compute the cost of a plan—in this case, the cost of an experiment. We define the cost of an experiment based on propriety information gathered from specific laboratories. This information includes both monetary and human costs.

Listing 2: Example SHOP2 cost computation.

```
(:op (!!calc-culture-cost ?performer ?meth)
    :precond
      ((cost ?meth
             ?performer ?cost-per-sample)
       (min-sample ?meth ?performer
                   ?number-of-samples))
    :cost (* ?cost-per-sample
             ?number-of-samples))
```

Listing 3 shows an example SHOP2 method for provisioning yeast colonies into samples to be used in an experiment. This is a recursive HTN method in SHOP2's language; it enables the planner to iterate over the yeast colony ?resources given in the first argument to the head task and provision a replicate sample for each of those resources. Unlike traditional planning model languages, SHOP2 allows sets as possible values for variables in a method or operator. For example in Listing 3, the variables ?resources and ?provisioned-samples hold lists of yeast colony descriptions.

Listing 3: Example SHOP2 method for sample provisioning.

```
(:method
 ;; Head task:
 (provision-all-resources ?resources
                   ?provisioned-samples)

 ;; preconditions for recursion base case:
 ((= ?resources nil)
  (bagof ?map (provision-sample-map ?map)
         ?sample-map))
 ;; subtasks(s) for the base case:
 (!provision :name ?name
   :transformations ?sample-map)

 ;; Recursive step
 ;; preconditions
 ((= ?resources (?resource . ?rest)))
 ;; subtasks
 ((provision-replicates ?resource
         ?type ?replicates)
  (provision ?rest ?provisioned-samples)))
```

The first precondition specifies the base case: checking to see if all of the resources have been provisioned, i.e., if the `?resources` list is empty. If so, the planner collects the entries of the provision sample map. SHOP2's preconditions language includes Prolog's `bagof`, which finds all bindings to a variable in a logical formula and collects them. Listing 3 collects the values of the `?map` variable from every grounded `provision-sample-map ?map` logical expression in the current state. Next SHOP2 will invoke the `!provision` operator with information from the sample map. In the recursive branch, the preconditions specify that the resources list must not be empty and split it into a first resource and the rest of the resources. The subtasks when this match are to provision the replicates for the first resource, and then recursively handle the remaining resources.

4 Value of Information

Providing predictable cost information was one of our sponsor's highest initial priorities: they are very concerned with facilitating scientific discovery by making labs-for-hire easier, more transparent, and more cost-effective to use. As we described above, XPLAN can already provide estimated costs for performing protocols at multiple labs. To go beyond this and provide further support to users, we are adding techniques, based on *value of information* (VOI), to estimate the benefit of particular protocols, so that XPLAN (likely in collaboration with its user) can guide users to more informative experimental protocols. We also hope that the analysis we conduct in the process will shed light on questions such as "how many biological and technical replicates are appropriate?", "how important is it to test *this* design across multiple laboratories?", and "how many tests are necessary to build confidence that a design is reliably replicable?"

In conventional decision analysis, VOI is defined as the difference between the expected utility of a decision made with a particular piece of information, and without that information (Pearl 1988, Chapter 6)). In design problems proper, we can use the estimated value of a successful design to compute the value of information that contributes to the design. For cases where the design is not directly useful (today many designs are made for exploratory reasons, not for employment), we will take the information produced (in terms of information distance between prior and posterior) as a proxy for utility. Unfortunately, VOI is notoriously difficult to compute (Krause and Guestrin 2009), because it requires reasoning about multiple possible outcomes of experiments.

Our work on this part of XPLAN is at a very early stage, but we can characterize the direction we are taking. We expect to use Monte Carlo Tree Search (MCTS) to approximate the value of information (Kamar and Horvitz (2013) use this technique but in a much simpler problem). Since the information-gathering process will be driven by execution of experimental processes, we will use SHOP2 to build the trees for the MCTS. We are still working out the extent to which

[3]See Appendix A.

the problem will involve conditional planning – typically there is little closed-loop control of the experimental protocol based on the information produced. That information is generally extracted in an offline data analysis process after the protocol is completed. Closed-loop control is typically limited to correcting failures. If true, that will simplify the construction of the protocol significantly, and avoid the need to generate a large and complex experimental *policy* instead of an experimental *plan*. That said, XPLAN will still have to explore many branches to find the VOI of alternative plans.

5 Conclusions and Future Work

Our XPLAN system uses HTN planning in an interesting new domain: experiment planning for synthetic biology. XPLAN exploits the expressive power of the SHOP2 planner to handle many of the challenges in coupled engineering design and scientific exploration of emerging fields. It adapts to domains with weak mechanical models, in a way that would be difficult, if not impossible for first principles planners. It has already shown utility by computing experiment costs across different labs, and by automating the process of aligning experimental data with experimenter intent in ways that enable the automation (and hence the speed-up) of data analysis. The fact that XPLAN's plans can be compiled into executable procedures will provide value in the near future, as the SD2 pipeline is completed. In ongoing work, we are extending our cost modeling to incorporate benefits – in terms of VOI.

A SHOP2 is Turing-complete

Since SHOP2 can invoke arbitrary functions in its preconditions (Nau et al. 2003), it can invoke a Turing machine simulator as an external function, and have a task network that would take a universal Turning machine program as parameter and return a plan iff that program terminates. □

References

Erol, K.; Hendler, J.; and Nau, D. S. 1994. UMCP: A Sound and Complete Procedure for Hierarchical Task-Network Planning. In *Proc. Second International Conf. on AI Planning Systems (AIPS-94)*, 249–254.

Goldman, R. P., and Kuter, U. 2018. Explicit stack search in shop2. Technical Report SIFT-TR-2018-1, SIFT, LLC, Minneapolis, MN.

Goldman, R. P. 2009. A semantics for HTN methods. In Gerevini, A.; Howe, A. E.; Cesta, A.; and Refanidis, I., eds., *ICAPS*. AAAI.

Kamar, E., and Horvitz, E. 2013. Light at the end of the tunnel: A Monte Carlo approach to computing value of information. In Gini, M. L.; Shehory, O.; Ito, T.; and Jonker, C. M., eds., *AAMAS*, 571–578. IFAAMAS.

Krause, A., and Guestrin, C. 2009. Optimal Value of Information in Graphical Models. *Journal of Artificial Intelligence Research* 35:557–591.

Madsen, C.; McLaughlin, J. A.; Mısırlı, G.; Pocock, M.; Flanagan, K.; Hallinan, J.; and Wipat, A. 2016. The SBOL Stack: A Platform for Storing, Publishing, and Sharing Synthetic Biology Designs. *ACS Synthetic Biology* 5(6):487–497.

McLaughlin, J. A.; Myers, C. J.; Zundel, Z.; Mısırlı, G.; Zhang, M.; Ofiteru, I. D.; Goñi-Moreno, A.; and Wipat, A. 2018. SynBioHub: A Standards-Enabled Design Repository for Synthetic Biology. *ACS Synthetic Biology* 7(2):682–688.

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *JAIR* 20:379–404.

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Muñoz-Avila, H.; Murdock, J. W.; Wu, D.; and Yaman, F. 2005. Applications of SHOP and SHOP2. *IEEE Intelligent Systems* 20(2):34—41.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Waltham, MA: Morgan Kaufmann.

Roehner, N.; Beal, J.; Clancy, K.; Bartley, B.; Misirli, G.; Grünberg, R.; Oberortner, E.; Pocock, M.; Bissell, M.; Madsen, C.; Nguyen, T.; Zhang, M.; Zhang, Z.; Zundel, Z.; Densmore, D.; Gennari, J. H.; Wipat, A.; Sauro, H. M.; and Myers, C. J. 2016. Sharing Structure and Function in Biological Design with SBOL 2.0. *ACS Synthetic Biology* 5(6):498–506.

Wilkins, D. E. 1988. *Practical Planning: Extending the Classical AI Planning Paradigm*. Morgan Kaufmann Publishers.