STATUS DIGEST # 4:
CADCAD AS A DECISION SUPPORT SYSTEM

DRAFT V. 0.8

BlockScience

June 2, 2021

OVERVIEW

The **cadCAD** representation of the STATUS network can be used to provide recommended ranges of parameters under the control of the network's designers, an engineering process known as *parameter selection under uncertainty*. Once the network is live and real data is generated, **cadCAD** can then be leveraged to provide 1) feedback regarding its existing state (via backtesting and counterfactual analysis), 2) forecasting of the future state of the network (by creating forward-looking trajectories), and 3) a living laboratory *digital twin* that uncovers potential ramifications of e.g. governance or protocol changes. As a digital twin **cadCAD** becomes a **Decision Support System** of the live STATUS network, on behalf of the network's designers, developers and stakeholders.

Contents

1 Introduction **2**

2 Parameter Selection Under Uncertainty **2**

2.1 System goal identification 3

2.2 Control parameter identification 3

2.3 Environmental parameter identification 4

2.4 KPIs and metrics 4

2.5 Scenario Selection 5

2.6 Experiment workflow and parameter range selection 6

3 cadCAD as a Decision Support System **7**

3.1 Backtesting 7
3.2 Counterfactuals 8
3.3 Trajectories 8
3.4 Interventions 9

1 Introduction

Once a **cadCAD** implementation of the system has been created (as usually set out from the system's Mathematical Specification) there are two different, but related, workflows that provide both guidance for and feedback to the system's actual, real-world implementation. The first workflow generates experimentation results that inform the initialization of parameters within the system, *prior* to its live instantiation. This workflow builds upon the modeling assumptions and parameterizations that characterize the *implementation* of the Mathematical Specification in **cadCAD**, and is referred to as **parameter selection under uncertainty**. This is the subject of Section 2. The output of the workflow, conditional upon the modeling assumptions and specifications of any external environmental processes, is a collection of ranges of parameter values for which the system goals of the project are achieved, as measured by Key Performance Indicators (KPIs) or other associated metrics that reflect these goals. The ranges act as 'basins' from which a selection of parameters will achieve a balance between system goals that the system's designers and/or stakeholders have specified (perhaps iteratively as results of the workflow become available). The parameters selected may be used 'as-is' or consumed as part of the stakeholder decision-making procedure, for the initialization of the larger system when it goes live.

The second workflow takes place *after* the system has gone live, either using or indirectly informed by the range of parameters provided by the parameter selection workflow. The live system generates actual data that is affected by (and in some cases estimates) otherwise private or external characteristics that impact the system, such as random processes or participant preferences. These characteristics may have had assumed values (or processes) under the first workflow, reflecting the overall modeling assumptions. With the advent of live data, however, the model assumptions may be refined, or omitted altogether, as the data provides an actual view of the responsiveness of the system to the external characteristics. This use of live data *evolves* the **cadCAD** specification into a *Decision Support System* (DSS), and is referred to as the creation and maintenance of a *digital twin* of the live system. This is the subject of Section 3.

2 Parameter Selection Under Uncertainty

The cadCAD workflow rests upon a standardized engineering template, which moves from the overall goals defined for the system to the eventual selection of parameters that meet criteria reflecting those system goals. This workflow partitions the space of possible system designs according to what can be directly implemented via design decisions (the "control parameters"), while providing an inference infrastructure for incorporating what cannot be controlled or

implemented (the "environmental parameters"). The ability to connect system goals to the selection process of control parameters given environmental parameters is mediated by defining key performance indicators (KPIs) or metrics. These allow system goals to be quantified and hence (when, as usual, there are multiple such goals) to be compared and balanced against one another.

The Parameter Selection under Uncertainty (PSuU) workflow outline is broken down into the following steps (each of which is described briefly in further detail below):

1. System goals are identified;
2. Control parameters are identified;
3. Environmental parameters are identified;
4. Key performance indicators (KPIs) / metrics are identified;
5. Simulations scenarios are identified;
6. Experiments are conducted; and
7. Optimal parameter ranges are selected.

2.1 System goal identification

System goals ensure the cadCAD simulation parameters and metrics are aligned with the overall objective. As described in the first Status Digest, "Architecture", the system goals for the STATUS project are:

1. A **Client** should prefer to route a **Message** to its final recipient as quickly and cheaply as possible;
2. A **Client** should prefer to store a **File**, and allow a **File** to be retrieved, if it is possible to do so;
3. Opportunities to "game the system", by manipulating **Client** decentralized incentives for idiosyncratic benefit, should be too expensive to exploit;
4. Value transfer (e.g. compensation for **Message** routing, and reward/penalty incentive payments) should be trustworthy, and explicit enforcement of incentives should be undertaken as infrequently as possible.

2.2 Control parameter identification

Control parameters specify those critical features that can be selected by system designers to achieve the system goals. For the STATUS project, the following control parameters have been identified in the previous Digests:

1. The functional form parameter k for the decreasing logistic function used in the trust update function, cf. STATUS Digest #2, Appendix A;
2. The audit rate for the **Contract** component, cf. STATUS Digest #2, Section 3.

2.3 Environmental parameter identification

Environmental parameters specify features that are external to the system, but affect the achievement of the system goals. For the STATUS project, environmental parameters impact the construction of the **Nodes** of the system, i.e. the connectivity and growth of the network. In addition, the creation of **Messages** to propagate through the network is governed by requests for **Files**, which is modeled as an exogenous process. An example of environmental parameters fulfilling these requirements might be:

1. The growth rate of an exponential random graph model of network development;
2. The arrival rate of **Message** requests from e.g. a Poisson process over **File** requests.

2.4 KPIs and metrics

Given environmental parameters, control parameters are selected in response to aggregations of KPIs/Metrics that reflect the system goals described in Sec. 2.1. The following metrics could be identified as providing a means to assess whether or not particular control parameter choices fulfill these goals. Note that these metrics are themselves usually then framed within *threshold criteria* that allow outcomes based upon alternative control and environmental parameter combinations to be ranked. Example threshold criteria are described after each metric identification:

1. **Timeliness**: The time taken for a **Message** to arrive at its final recipient. Threshold criterion: *the fraction of **Messages** that take longer than a given amount of time to arrive must be less than a given percentage*;
2. **Arrival**: The number of **Files** that were not stored / the number of **Files** that were not retrieved upon request. Threshold criterion: *The network-wide fraction of **Files** either not stored or not retrieved upon request, per given time interval, must not exceed a given percentage*;
3. **Productivity**: The number of self-dealing **Clients** in the network. Threshold criterion: *The fraction of self-dealing **Clients** in the network with at least a given fraction of their trust scores strictly greater than zero must be less than a given percentage*;
4. **Enforcement**: The number of audits per time interval. Threshold criterion: *The audit 'hit rate', i.e. the fraction of audits confirming self-dealing behavior, must not fall below a given percentage.*

Even a cursory perusal of the metrics defined above indicates that metric changes improving one threshold criterion will not necessarily improve *all* threshold criteria. For example, having a high audit rate may reduce the number of self-dealing **Clients** to zero, satisfying the third threshold criterion, but in the process makes it likely the fourth criterion is violated (as there are then "too many" audits). As a result, one should generally expect that the process of parameter selection will lead to *tradeoffs* between system goals, as exemplified by their associated metrics and threshold criteria. As the complexity of the system increases, such tradeoffs become the norm rather than the exception, and the PSuU workflow must have assessment systems in place that provide stakeholders with an understanding of the tradeoff 'landscape'.

One way to visualize these tradeoffs is to leverage the workflow to *classify* the resulting experiments according to how well one (or more) parameter values satisfy one (or more) system goals. As a simple demonstration, suppose that there exist three system goals–this can be visualized as a simplex (for three system goals, a triangle) with vertices representing system goals and edges between vertices indicating the relative frequency of experimental results supporting one goal over another. Interior points are then relative weights between all three system goals, providing a mapping from the swept parameter space to the relative weights (and hence, to the relative frequency of experiments supporting system goals). Such a visualization is given in the abstract in Figure 1.

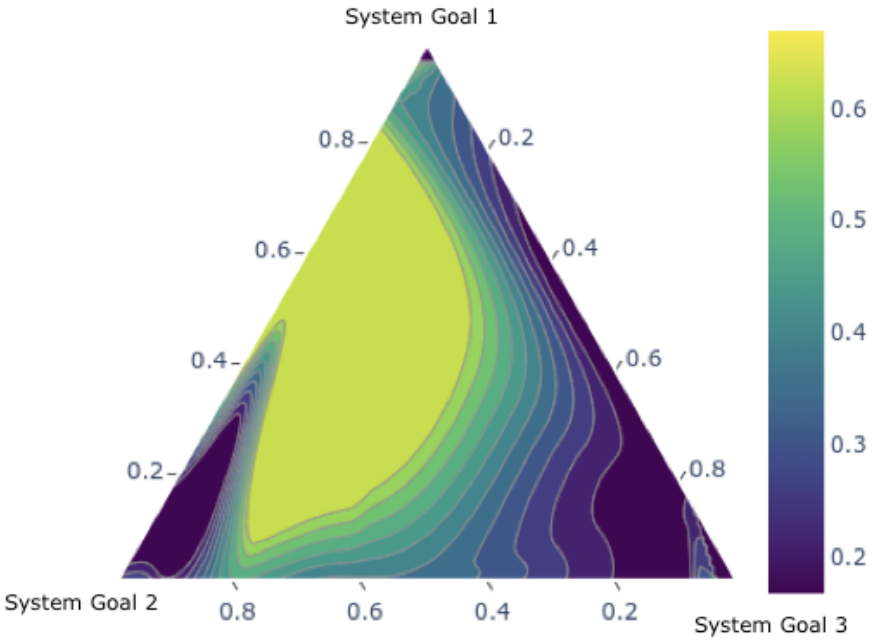

Figure 1: Parameter Basin of Attraction, Balancing System Goals

2.5 Scenario Selection

When control and environmental parameters have been specified, and the metrics/KPIs and threshold criteria associated with system goals have been developed, the various scenarios to be simulated must be articulated. Such scenarios typically commence with *sanity checks*, where the environmental parameters are set to values for which the system dynamics are well-understood, and there are *a priori* expectations of simulation outcomes conditional upon various

control parameter values. During sanity checking, less attention is given to threshold criteria and more to basic overall measures, such as system stability and internal consistency.

After sanity checking, scenarios often proceed with *structured exogenous shocks*, in which environmental parameters are set to provide a series of "input signals" that the system processes. The response of the system to these input signals can be incorporated into the sanity check workflow, to test system robustness, or used to assess one or more threshold criteria. In the latter use case, shock testing can help to narrow the neighborhood of control parameters to search over, by identifying areas of the control parameter space that respond better to shocks (in terms of both system stability and threshold criteria) than others. Future parameter searches can then be focused upon those areas, saving simulation and computation time.

2.6 Experiment workflow and parameter range selection

Once the simulation scenarios have been created, the full experiment workflow can be performed using **cadCAD** to select a range of control parameters according to the KPI/metric threshold criteria. The general schematic for a series of experiments is:

1. Specify the generating process(es) for environmental parameters: can be ranges, distribution assumptions, historical fit, etc.;
2. Specify the sweep ranges for control parameters;
3. Perform **cadCAD** experiment simulations for each scenario across sweep ranges, using Monte Carlo numerical analysis (for stochastic draws) and synthetic shock tests;
4. Assess the KPIs/metric threshold criteria for each experiment outcome using e.g. ranking, machine learning, system goal landscape mapping, qualitative analysis etc.;
5. Select the candidate parameter ranges based upon this assessment;
6. Perform a robustness/sensitivity analysis within the candidate parameter ranges; and
7. Derive optimal parameter ranges.

At the conclusion of this process, the results are recommended *ranges* of parameter values for the identified control parameters. These ranges can then be used by stakeholders to inform system design decisions prior to going live. In the STATUS project example, the ranges of trust function parameter values and audit rates provide a suggested implementation of the incentive mechanisms, as outlined in Status Digest #2.

It is possible that a stakeholder would decide that the purpose of the engineering process has been completed at this point. Following the previous Digests, the architecture and the incentive mechanisms have been defined, the **cadCAD** specification has been developed and subsequently used to inform system designers of how to set certain free (control) parameters, and this may conclude the original project objective. In most cases, however, the provision of control

parameter ranges is the point of departure for using **cadCAD** as a means to help understand the system *once it has gone live*. This is the subject of the following Section, where **cadCAD** as a Decision Support System (DSS) is discussed.

3 cadCAD as a Decision Support System

The DSS workflow is implemented through **cadCAD** in its role as a *Digital Twin* of the existing live system. The output of this workflow depends upon the degree to which live data is used to inform different aspects of the pre-live **cadCAD** system, and may be broken down into four 'stages' with progressive dependencies:

1. Backtesting;
2. Counterfactuals;
3. Trajectories; and
4. Interventions.

Each part of the workflow is briefly described below.[1]

3.1 Backtesting

First, and at its most basic, live data can verify that the pre-live **cadCAD** system behaves in the same way that the live system behaves. This allows one to examine the properties of the **cadCAD** implementation without a subset of modeling assumptions that were made to create parameter range recommendations as output from the PSuU workflow. This is akin to 'turning off' different **cadCAD** implementations of components and/or modules, creating a 'short-circuit' that uses live data to represent the required input-output relationship the module(s) represented. Such components are normally 1) models of stochastic generating processes that depend upon the environmental parameters assumed, and 2) models of participant deliberative behavior, that depend upon *assumed* characteristics.

For example, in the STATUS **cadCAD** representation, the trust incentive mechanism between peer-to-peer (p2p) **Clients** is encapsulated into a module that utilizes a trust score mechanism (cf. Digest #2). The actual trust scores, and indeed the actual incentive behavior, of p2p **Clients** is likely unobservable in real data, however. In order to test whether or not the model is *consistent* with live data, one may remove the trust mechanism from the **cadCAD** implementation and replace it with the inputs to the mechanism, e.g. actual neighborhoods and incoming **Message** traffic, and the resulting outputs of the mechanism, e.g. outgoing **Message** traffic, and see if the resulting system dynamics replicate the actual behaviors in the live data.

In essence, backtesting replaces an *input-module-output* implementation in **cadCAD** that was used for the PSuU workflow with an *input-output* relationship taken directly from real data, to assess how well the remaining modules in the implementation replicate the behavior seen in the live system.

[1] Note that, as the STATUS project concluded before the commencement of the PSuU workflow, what follows is treated in a more abstract framework.

3.2 Counterfactuals

Counterfactual analysis is closely related to backtesting: instead of replacing an *input-module-output* implementation with an *input-output* relationship, different modules can be *substituted* for those selected prior to the PSuU workflow, and assessed for their ability to explain the live system data better than the previously selected modules. In other words, *input-module A-output* is replaced with *input-module B-output*, where *module A* is the original implementation in **cadCAD**, and *module B* is the alternative module. This allows one to test various 'what if?' scenarios that experience with the live data might suggest are worth examining.

In the STATUS **cadCAD** representation, instead of 'short-circuiting' the incentive mechanism (as described for backtesting), one might decide based upon live data that p2p **Clients** are behaving according to another incentive mechanism than the trust score mechanism assumed. In that case, one would model this alternative mechanism and replace the existing trust score mechanism, and then proceed to examine the **cadCAD** system dynamics on real data. The latter examination is usually facilitated at this stage by the backtesting results, where e.g. stochastic generating processes have already been replaced with live data.

3.3 Trajectories

Backtesting and counterfactual creation use live data as a way to test system stability and performance by either removing pre-live **cadCAD** implementation modules, or by implementing alternative modules that have been specified as *a priori* 'what if?' scenarios. But it is also possible to *create* such implementation modules from the data itself. This enhancement immediately extends the capabilities of **cadCAD** to *forward looking* extrapolation, projection and prediction. Such implementation modules are derived from statistical analysis and/or Machine Learning (ML) techniques, by examining relationships within the live data and extracting those relationships as a basis for the projection of system trajectories. These trajectories in turn inform stakeholders about possible ranges of system parameters that are consistent with the most recently available data onwards, with fidelity that (depending upon the assumptions of external sources of uncertainty) may also provide statistical range measures such as trajectory confidence intervals.

For example, an ML 'black box' could be fit to match the evolution of the STATUS network's 'active **Clients**', i.e. mapping an existing real-world state of the network restricted to **Clients** actively sending **Messages** (perhaps over some time period) to the future real-world state of the network with the same restriction. Such a **system identification** methodology learns a (current network state) -> (future network state) relation using characteristics of the network, such as **Message** arrival rate, audit rate, etc. as *features*. The resulting black box can then replace the explicit trust incentive mechanism as in the backtesting and counterfactual workflows, but with the extended ability to allow *extrapolation* and *prediction* to take place. Over time, as the features of the system relevant to the ML methodology change, the black box relation can be updated (either incrementally, or over batches of new data). This allows forward-looking trajectories to be created using the most recent data available that is computationally feasible to integrate into the existing relation.

3.4 Interventions

Finally, with the backtesting-counterfactual-trajectory workflow in hand, it is possible to introduce into the **cadCAD** implementation alternative governance decisions, system updates/upgrades or other aspects that change the underlying system dynamics (the "rules of the game") in some fundamental fashion. Such changes act as **interventions** into the system, and assessing the impact of alternative interventions provides both 1) information on e.g. future trajectories conditional upon an intervention, and 2) a characterization of the timing of such interventions or even the structure of their implementation. As the most complex usage of real data, intervention analysis builds upon the entire digital twin workflow and completes the evolution of **cadCAD** into a representation that runs side-by-side with the live system, continuously providing a 'dashboard' of metrics, statistics and trajectories that provide a snapshot of system health in the future, as well as the state of the system under novel (and perhaps unforeseen) changes to the system's underlying dynamics.

Within the STATUS network, such a change in the 'rules of the game' might be how the **Contract** component interacts with the p2p **Client** component to disseminate audit results. Rather than broadcasting the result of a 'bad' audit to all **Clients**, suppose that an alternative is being considered by stakeholders wherein the **Contract** layer is actively empowered to slash staked collateral from each **Client** as part of a participation requirement in the network. This is a significant change to the incentive structure, and would be very difficult (if not impossible) to test using a live, working system without large-scale disruptions of both system performance and user expectations (and hence participation).

By contrast, using the digital twin workflow, creating a module with this new punishment mechanism is straightforward—backtesting, counterfactuals and especially trajectories can all be built around this proposed change in the system, and the resulting system dynamics can inform stakeholders at the governance level as to whether or not adopting the new mechanism is preferred. This also allows, at the same time, the incorporation of new KPIs/metrics (and associated threshold criteria) that either become more important as the system evolves, or are introduced as entirely new measures *because of the observations of the live system*. This is a strength of the digital twin, as it can flexibly adapt to new stakeholder preferences that are only revealed over time, rather than 'fixing in stone' the desired performance metrics prior to the advent of the network's real-world implementation.