

Equation-Based Policy Optimisation for Agent-Oriented System Dynamics Models¹

Jim Duggan,
College of Engineering & Informatics,
National University of Ireland, Galway.

Note: This is a pre-refereeing pre-print which adheres to Wiley-Blackwell's Open Access publication policy. The link to the published article is:
<http://www3.interscience.wiley.com/journal/119163362/abstract>

1. Introduction.....	2
2. Problem Characterisation.....	4
3. System Design	5
4. Case Study	8
5. Conclusions.....	13
6. References.....	13
A. Appendix 1: Beer Game Optimisation Characterisation	15
B. Appendix 2: Beer Game Network Model	18
C. Appendix 3: Beer Game Sample Strategy File	21

ABSTRACT

While System Dynamics is normally associated with the representation of high-level dynamic aggregate models, it has also been successfully applied to modeling agent-based systems. Static networks of cooperative agents can be constructed, and their interactions modeled through equation assignments from one element of a model to another. Within System Dynamics, optimisation has played an important role in defining the best policies for any given model: however, traditionally, these optimal solutions are based on finding the best combination of model parameters that maximise or minimise a payoff function. This paper presents an optimisation approach that can explore a search space in order to discover the best combination of parameters and equation-based strategies for a given agent-based problem. The approach is validated through a case study based on the four-agent Beer Game.

¹ The software system presented in this paper runs on the .NET platform, and is freely available from the author.

1. Introduction

In recent years, considerable interest has been shown in exploring synergies between system dynamics and agent-based simulation, and it is recognised that both approaches have the capacity to deliver overlapping and complimentary insights into complex problems (Chaturvedi et al. 2001, Choi et al. 2001). For example, in the domain of corporate networks dynamics, Akkermans (2001) presents a system dynamics model of 100 agents, and focuses on the question of whether such corporate networks can achieve stability. His analysis indicates that “lock in” emerges in the supply network, and that agents that bias their strategy towards short-term performance outperform agents that adopt longer term business relationships. A key point of this research is that the agent perspective incorporating state, feedback, and inter-agent interactions can be represented in system dynamics, in terms of models of disaggregated integral equations.

Borshchev et al. (2004) show how several agent based models can be constructed from existing system dynamics models. Examples include the classic predator-prey model, and the Bass diffusion model which simulates the rate of adoption of new products. Scholl (2001) reflects on how the emergence of complexity science, and its primary focus on agent-based modeling, effectively overlooked the rich body of literature from system dynamics. He argues for cross study and closer ties are long overdue, and that one practical way forward would be to develop agent-based models of system dynamics classics such as the Beer Game. At a technical level Grossler et al. (2003) demonstrate how both approaches can be integrated, and RePast® (North et al. 2006) and Vensim®² are used to model different actors in a supply chain. Furthermore, the authors suggest that future promising areas for synergy between the two approaches include complex problem situations such as the dynamic reconfiguration of a supply chain’s structure, and determining the effects of changes on overall performance.

The idea of combining system dynamics and agent simulation to dynamic supply chain configuration is also explored by Schieritz and Grossler (2003). They contend that the combination these approaches reduces the a priori complexity of supply chain models, through representation of the micro-level agent schemata by system dynamics models, while the higher supply chain network is modeled using a conventional agent approach. In this scenario, the phenomenon of agents deciding between suppliers can be accommodated, as individual agents maintain mental models (stocks) of key decision variables such as supplier attractiveness and perceived delivery time, and the high-level agent environment can reorganise the agent network based on the lower level preferences. As the authors explain, this level of flexibility would not be possible in a conventional system dynamics model, where the structure is determined before commencing the simulation, and cannot be altered during the course of a given simulation experiment.

² RePast and Vensim are registered trade marks.

Another study which employed both approaches is one that involved the analysis of cellular receptor dynamics (Wakeland et al. 2004). The authors explain that these types of problems are frequently tackled using differential equations, and hence the suitability of system dynamics. However, in certain scenarios where concentration and reaction probabilities are low, agent-based modeling does offer benefits. In their paper, they demonstrate how both approaches can be used, although they did not find a clear demarcation which would indicate which technique is preferable in a given situation. Rahmandad (2004) carries out an extensive analysis of the spread of contagious diseases, using system dynamics and agent-based approaches. The impact of heterogeneity in agent attributes, and the effect of different network structures are studied, and the work demonstrates that agent-based and differential equation elements can be combined in the same model. In referring to Forrester's seminal work on Industrial Dynamics (Forrester 1961), and in common with Scholl (2001), who commented on the lack of awareness of the complexity community of this body of work, Rahmandad writes that "one of Forrester's key contributions to modeling methodology was precisely to view systems as comprised of multiple interacting agents, each perceiving only a limited subset of all available information, and each with their local goals, norms and decision rules." This view, and indeed the work cited in this paper, clearly confirms that a synergy exists between the two modeling methodologies.

An important dimension of system dynamics, and indeed any simulation approach, is the use of optimisation in order to traverse the policy space in order to find the best combination of parameters, based on a specified objective function. The system dynamics literature, for example, Coyle (1996), Dangerfield et al. (1996), Chen et al. (2004), Duggan (2005), Keloharju and Wolstenholme (1989), describes how optimisation and simulation are complimentary approaches, and how, when used as part of the modeling process, optimisation can impart key understandings to decision makers. A common thread through these papers is what could be termed the *parameter perspective* for optimisation approaches. For this type of scenario, it is assumed that the policy maker is satisfied with the equations, and the only doubt remains as to what values the parameters – or the policy levers – should have. This is an important problem to solve, as in many systems, different parts of the parameter space can give rise to huge variances in the value of a payoff function. However, these optimisation approaches do not explicitly allow policy makers to experiment with different equation structures in addition to exploring the parameter space.

What this paper now proposes is what could be termed the *equation perspective* to optimisation, which, in addition to allowing for the inclusion of parameters for an optimisation problem, also allows the decision maker to allow for the evaluation of many different sets of heuristics (i.e. equations) which agents can then employ in order to arrive at the optimal set of parameters *and* policy equations for a given problem. The general problem is characterised, the system design described, and a case study based on the Beer Game is presented.

2. Problem Characterisation

The type of problem addressed by this work (see figure 1) may be characterised by:

- A. A network of N agents (A_1, A_2, \dots, A_N), represented as a bi-directed graph, where each agent A_j can be connected to a predecessor agent A_{j-1} and a successor agent A_{j+1} . Agents that have no predecessor or no successor can also be represented.
- B. A set of M agent strategies (S_1, S_2, \dots, S_M), and each of these can be deployed by any of the N agents. Each strategy S_i (i.e. each agent's behaviour) is defined by a set of stock and flow equations.
- C. A set of agent parameters (PA_1, PA_2, \dots, PA_N), which are identified as policy "levers" than are to be modified in order to seek out the optimal solution for a given payoff function. The number of parameters for each agent is defined as P .

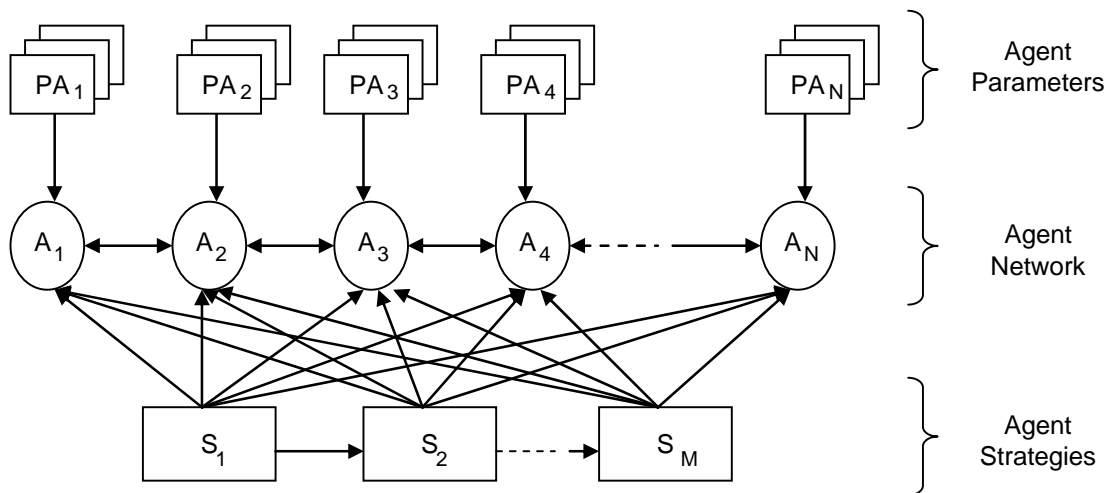


Figure 1: Characterisation of the optimisation problem

Given these definitions, for a given agent network whose structure can be defined as a bi-directed graph, and whose behaviour is specified through stock and flow equations, the goal of the system is to find the set of parameters and strategies (equations) that optimises the target payoff function.

3. System Design

The structure of the system is similar to how optimisation is usually approached in system dynamics, namely optimisation through repeated simulation (Coyle 1996), where sets of parameters are passed through to the optimiser, and the payoff function value then returned. This result is then used to assess the suitability of the parameter set, and the optimisation process continues until a solution converges on an optimal value.

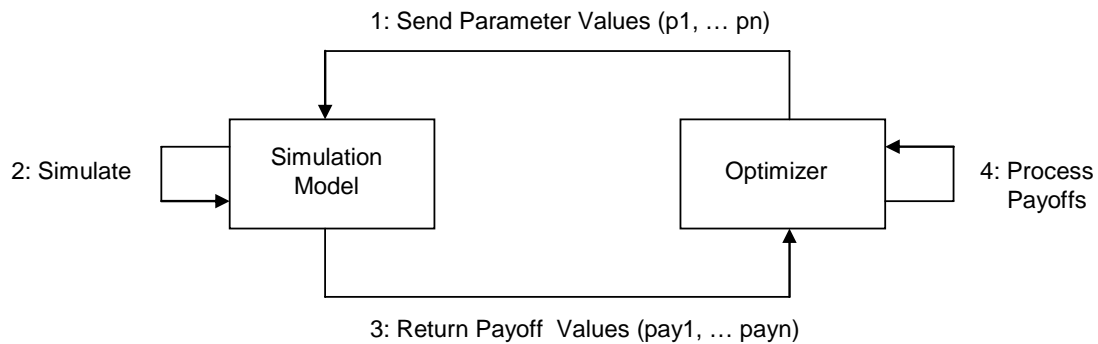


Figure 2: The optimisation process in system dynamics

For this system, a purpose-built genetic algorithm and simulation software is used, which is an extension of software already developed to discover Pareto-optimal solutions for system dynamics models (Duggan 2006). An example a solution chromosome for this equation-based optimiser is shown in figure 3, where the size of the solution is defined as:

$$\text{SIZE}(\text{Solution}) = (N \times P) + (N)$$

where N is the number of agents, and P is the number of parameters per agent.

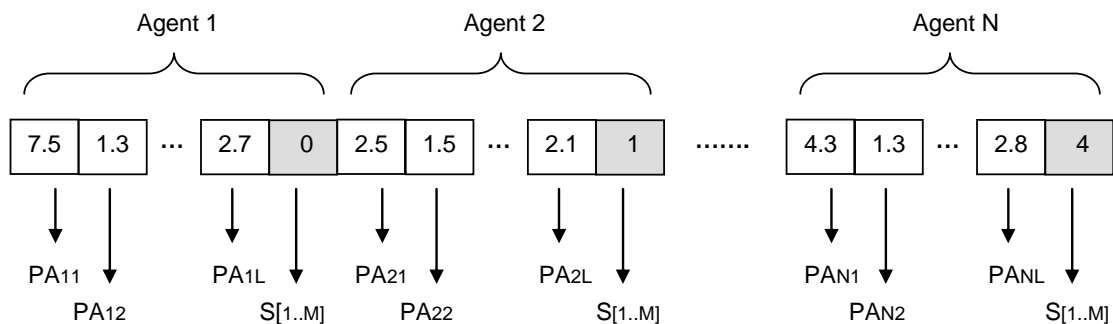


Figure 3: Solution Structure

Figure 3 represents a solution that has N agents, and there are P parameters for each agent. These parameters have a defined minimum and maximum value, and the optimiser generates values with those specified ranges. The additional N cells which are added to the solution represent the strategy identification number used by an individual agent, and will have a value of $[0.. M-1]$, where M is the total number of strategies.

The actual algorithm is standard, and roulette wheel selection with a ranking-based approach is used:

```

Randomise Population ( $P$ )
While ( $GenerationCount < NumberGenerations$ )
    CalculatePayoffs( $P$ )
     $MP = Selection(P)$ 
    Crossover( $MP$ )
    Mutate( $MP$ )
     $P = MP$ 
End While

```

The most complex element of the design now lies in preparing the solution so that a payoff can be calculated. While the values of each solution can be easily generated, based on selecting random values from the parameter range and the allowed set of strategy values, the follow-on challenge is to translate this array into a set of stock and flow equations which models the particular set of strategies for a given agent. Figure 3 illustrates how this is accomplished, where the inputs to the model builder are: the solution chromosome, the addition to a network model (see appendix 2), and a set of strategy equations (see appendix 3).

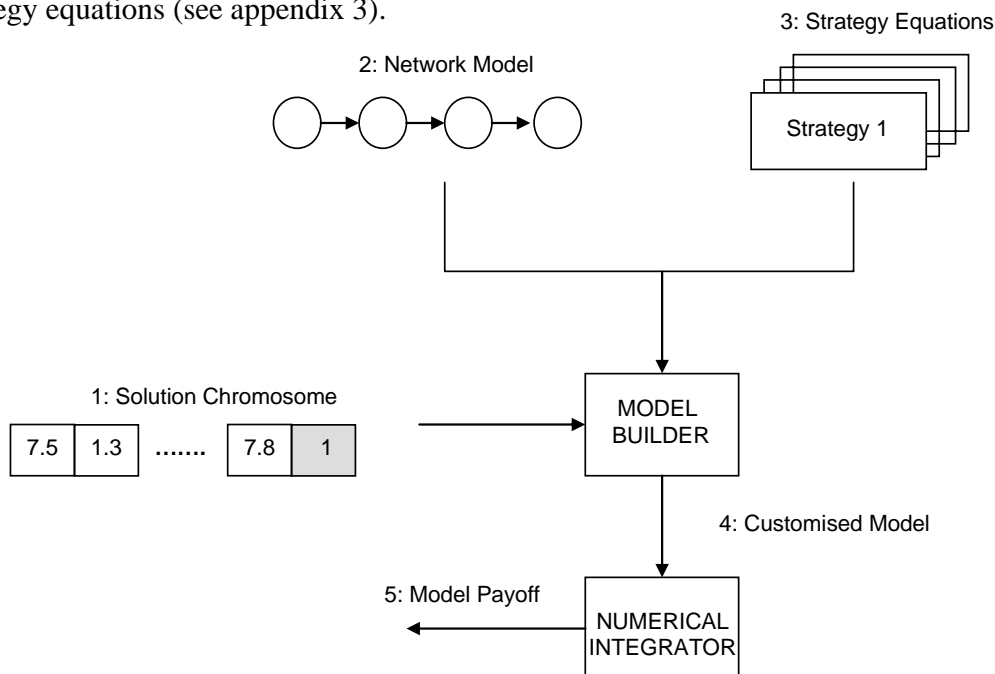


Figure 3: Steps involved in calculating payoffs

The payoff calculation is achieved using the following steps:

1. The input is a solution chromosome, which contains the “genetic information” that is needed to configure a simulation run. This information includes the values for each parameter, and an identifier for each agent’s strategy.
2. The model builder loads (1) the *network model*, which specifies, in the form of a bi-directed graph, the agents in the network and their interrelationships (see appendix B), and (2) all of the strategy files, which contain the sets of equations that capture each decision making rule that will be used as part of the optimisation.
3. Based on the strategy number specified for each agent, the appropriate strategy model file is dynamically loaded and combined with the network model to produce a single multi-agent stock and flow model. This stock and flow model will have the parameters and strategy set to those values specified in the chromosome. Mapping information is contained within the network file which maps both sides of an equation that involves information from both agents. For example, in the retailer agent specification, the variable *\$ID\$.UpstreamShipments* has one candidate for assignment, and that is the variable *\$ID\$.Downstreamshipments* in the wholesaler. [Note that \$ID\$ is a tag that is replaced by the actual ID of the agent in question.]

```
<candidates>
  <variable>$ID$.UpstreamShipments</variable>
  <candidate>
    <node_id>Wholesaler</node_id>
    <variable_id>$ID$.DownstreamShipments</variable_id>
  </candidate>
</candidates>
```

The mapping is important, as it allows a completely integrated model with consistent and valid equations to be created for each payoff calculation.

4. The newly generated stock and flow model is forwarded to a numerical integrator, and the payoff variables are then made available to the optimiser. The value of the payoffs are then used to rank individual solutions, and the standard operators of selection, crossover and mutation are then applied.

The optimisation process concludes when the defined number of generations have been reached. The highest ranked solution is the optimal, and this specifies the optimal combination of parameter values and strategies.

4. Case Study

In order to verify this approach, a case study based on a simulation of the Beer Game (Sterman 2000) is presented. The overall structure of the game is shown in figure 5 (see sample equations in appendix 3), where each agent sends orders upstream, and subsequently receives shipments after a time delay. In the Beer Game, each agent has a decision making heuristic based on the stock management structure, which formulates their order rate based on the current stock levels, the amount ordered but not yet arrived, and the expected orders.

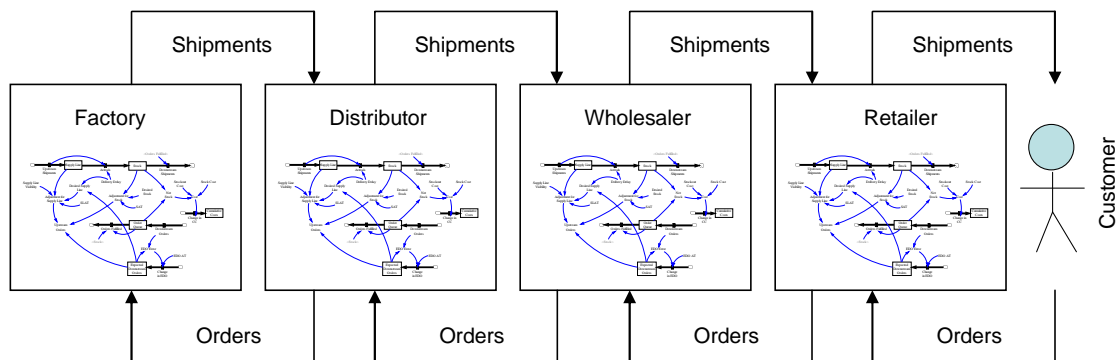


Figure 5: The Beer Game Network

An important educational aspect of the Beer Game is how this decision making structure of agents influences the overall system behaviour. It has been demonstrated (Sterman 1989) that when students play the Beer Game, they often use a policy that ignores the supply line, and so their ordering patterns are unnecessarily amplified, and this amplification increases in magnitude as the demand signals make their way upstream, giving rise to the well-documented “bullwhip effect.” In our scenario, in order to illustrate how the system operates, we present a range of different strategies which are based on a combination of the information cues used for the stock management structure.

These strategies are summarised in table 1, and each strategy presents a different heuristic which determines how much an agent orders at any point during the simulation run. For this example, six strategies were selected, but this value is an arbitrary selection, and in practice, there is no significant upper limit on the number of strategies that can be used. While some of these strategies might appear naïve, it is probable that they could be used in any given Beer Game scenario. Also, the goal of the case study is not to discover the best heuristic for the beer game, rather, it is to demonstrate that the proposed optimisation approach can be implemented for a given agent network.

ID	Formulation of order rate	Comments
0	$\text{MAX}(0, \text{ExpectedOrders} + \text{StockAdjustment} + \text{SupplyLineAdjustment})$	Correct stock management structure formulation
1	$\text{MAX}(0, \text{ExpectedOrders} + \text{StockAdjustment})$	Ignores supply line – models misperception of feedback
2	$\text{MAX}(0, \text{StockAdjustment})$	Stock adjustment only: ignores supply line and expected demand.
3	$\text{MAX}(0, \text{ExpectedOrders})$	Expected orders only: no consideration of current stock or supply line.
4	$\text{MAX}(0, \text{ExpectedOrders} + \text{SupplyLineAdjustment})$	Expected orders and supply line adjustment are considered.
5	$\text{MAX}(0, \text{SupplyLineAdjustment})$	Only the supply line adjustment is taken into account: expected orders and stock adjustment are ignored.

Table 1: The six ordering heuristics used for the experimental run

For each of the four agents, the following parameters, and their associated range, are considered:

- Expected Orders Adjustment Time (1, 15) – the adjustment time used to smooth out the actual orders using a first order information delay.
- Stock Adjustment Time (1, 15) – the adjustment time used to decide how many units are needed to move the current stock level towards its target value.
- Supply Line Adjustment Time (1, 15) – the adjustment time deployed to control how quickly the supply line approaches its target value.

In summary, for each agent there are four levers to consider: the three adjustment times, and the choice of strategy. In order for the optimisation to work, each of the strategies are coded into six different *plug and play* models, which can be configured in any combination by the optimiser in order to converge towards the optimal solution.

Figure 6 shows the initial conditions for the experimental optimisation run, namely:

- The problem characterisation, which includes the payoff function, the optimisation task, the genetic algorithm parameters and the customer demand function.
- The strategy files, which encapsulate the set of stock and flow equations that capture a particular strategy.
- The constraints for each parameter, and strategy, in the model. While values for the parameters can have fractional components, the values generated by the strategies will always be rounded to a whole number.

(1) PROBLEM CHARACTERISATION

PAYOFF: Global.TotalCosts
 TASK: Minimisation
 GENERATIONS: 100
 POPULATION: 100
 CROSSOVER RATE: 0.70
 MUTATION RATE: 0.15
 CUSTOMER DEMAND: 100 + STEP(100, 5)

(2) STRATEGIES

ID	Model
00	BG_00_TemplateSupplyLine.xml
01	BG_01_TemplateNoSupplyLine.xml
02	BG_02_TemplateASOnly.xml
03	BG_03_TemplateEOOnly.xml
04	BG_04_TemplateSLEO.xml
05	BG_05_TemplateSLOnly.xml

(3) RANGES FOR PARAMETERS AND STRATEGIES

	MIN	MAX
EOAT	1	15
SAT	1	15
EOAT	1	15
ST ID	0	5
<u>FACTORY</u>		
EOAT	1	15
SAT	1	15
EOAT	1	15
ST ID	0	5
<u>DISTRIBUTOR</u>		
EOAT	1	15
SAT	1	15
EOAT	1	15
ST ID	0	5
<u>WHOLESALE</u>		
EOAT	1	15
SAT	1	15
EOAT	1	15
ST ID	0	5
<u>RETAILER</u>		

Figure 6: Initial conditions for experimental optimisation run

In order to get a good range of results, twenty five optimizations were run, and the results of these are presented in table 2.

RUN#	RETAILER				WHOLESALE				DISTRIBUTOR				FACTORY				PAYOFF
	SAT	SLAT	EOAT	SID	EOAT	SAT	SLAT	SID	EOAT	SAT	SLAT	SID	EOAT	SAT	SLAT	SID	
1	8.47	11.21	9.52	1	6.66	14.53	3.84	1	1.88	2.23	2.67	0	3.02	7.05	1.55	1	60,355
2	6.67	2.89	5.44	3	11.78	4.8	2.72	0	2.53	9.03	4.62	0	2.38	12.77	1.25	0	68,645
3	11.15	14.65	3.53	0	8.45	14.4	2.72	0	12.99	1.87	1.15	0	1.88	9.16	3.6	1	51,468
4	6.22	13.43	6.18	3	4.71	1.66	2.96	0	8.26	7.28	1.76	0	2.65	9.43	2.01	1	68,040
5	11.06	3.44	1.23	3	8.07	7.26	2.79	0	2.2	4.65	5.45	0	1.74	2.74	1.05	1	54,009
6	14.57	11.1	2.11	1	7.12	1.03	2.93	1	3	1.08	2.73	1	1.45	7.56	1.28	1	55,262
7	10.93	1.19	6.24	1	3.73	6.75	1.4	4	3.44	2.12	4.58	0	7.27	11.88	1.69	0	76,171
8	7.21	6.59	14.08	1	5.96	5.14	3.55	0	11.95	1.55	1.06	0	1.68	6.78	7.82	1	57,751
9	8.36	6.91	9.37	1	7.21	13.96	1.4	1	7.62	5.93	1.9	0	1.85	3.11	1.77	1	58,682
10	7.21	11.86	14.38	0	1.57	10.06	6.99	1	4.88	11.27	6.95	1	1.19	14.16	10.25	1	133,028
11	13.05	11.53	6.48	1	10.84	2.72	3	1	2.6	14.74	4.92	0	1.17	6.88	9.04	1	64,917
12	13.52	12.13	1.8	4	6.32	2.94	2.62	0	14.77	9.92	1.59	0	3	4.97	1.91	0	64,463
13	8.96	2	2.74	3	7.14	11.22	7.82	1	9.02	5.83	1.15	0	2.3	6.87	2.35	1	71,833
14	10.26	14.06	11.29	0	6.76	10.56	1.31	1	13.71	6.35	1.87	0	5.47	6.36	1.93	1	74,686
15	12.14	1.78	11.45	4	7.55	1.17	2.74	0	2.02	2.53	14.84	0	1.52	9.48	8.36	1	79,018
16	10.93	11.72	1.56	4	5.88	7.91	6.84	1	2.98	1.23	1.52	0	3.03	8.83	1.12	0	54,797
17	12.11	6.42	7.52	1	6.62	11.4	5.72	0	3.71	7.39	3.41	1	5.04	10.73	1.7	0	78,336
18	2.44	9.76	7.77	3	9.96	6.06	1.94	0	10.11	9.8	1.87	1	2.65	12.14	3.95	1	77,622
19	12.09	6.83	3.21	3	12.69	13.89	1.88	0	7	1.19	1.55	0	3.72	5.73	2.71	1	57,920
20	7.54	4.76	1.87	3	2.45	7.81	14.82	0	7.9	1.8	1.93	1	1.17	10.01	8.84	1	68,730
21	13.74	5.39	4.52	0	5.54	9.02	1.08	0	7.46	11.98	2.27	0	9.52	4.63	1.03	0	69,471
22	3.98	5.37	14.46	1	11.78	7.37	1.39	1	3.81	4.56	1.65	1	9.59	1.14	1.39	0	62,375
23	7.21	9.46	1.35	3	9.85	9.51	2.13	0	2.24	4.04	8.23	1	1.76	1.8	1.12	0	54,888
24	13.61	5.89	3.67	1	3.62	7.67	8.61	1	3.49	12.92	1.77	1	1.37	3.04	1.23	0	54,143
25	7.78	1.74	9.06	3	13.12	12.89	1.09	0	14.75	8.03	3.19	1	11.25	6.65	1.59	0	92,893

Table 2: Summary of optimisation results

Figure 8 captures, over all the optimal solutions, the percentage of individual strategies used by each agent. The ones most prominent are those with ID 0 (use supply line) and ID 1 (ignore the supply line). While intuitively the reader may expect that the first strategy should win out for most cases, this is clearly not the case. Part of the reason for this is the complex interrelationships in the Beer Game, and also the effects that the adjustment times can play in “damping out” poor decision making. Furthermore, for this particular optimisation the simulations started in steady state equilibrium, and only one demand pattern was considered, which was the classical step increase in orders after five time units.

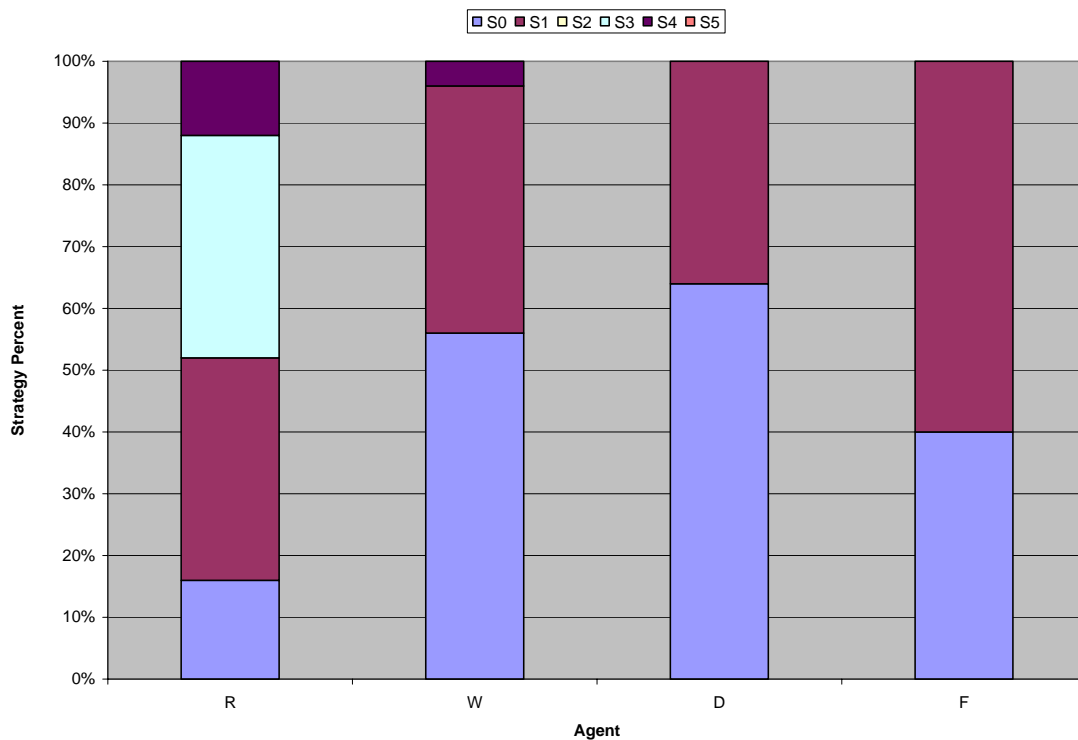


Figure 8: Breakdown of strategy adoption by each agent, for all recorded optimal solutions

5. Conclusions

This paper demonstrated a new optimisation method that can discover the optimal combination of parameters, and equation-defined strategies, for a given system dynamics agent-based problem. Such a system has the potential to be used in larger-scale agent-type problems, where the challenge is to test a range of heuristics in order to find the best one for a given scenario. Future work will involve the construction of a user-interface to augment the current file-based definitions. This will allow for: the construction of agent networks; the definition of agent behaviour; and the analysis of the optimisation results.

6. References

- Akkermans, H. 2001. "Emergent Supply Networks: System Dynamics Simulation of Adaptive Supply Agents." *Proceedings of the 24th Hawaii International Conference on Systems Sciences*, ISSN 0-7695-0981-9/01.
- Borshchev A. and A. Filippov. 2004. "From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools." *Proceedings of the 22nd International Conference of the System Dynamics Society*. Oxford, England. 2004.
- Chaturvedi, A., Dickieson, J., Daniel R. Dolk and J. Scholl. 2001. "Introduction to Agent Based Simulation and System Dynamics Minitrack." *Proceedings of the 24th Hawaii International Conference on Systems Sciences*, ISSN 0-7695-0981-9/01.
- Chen, Yao-Tsung, and Bingchiang Jeng. 2004. "Policy Design to Fitting Desired Behaviour Pattern for System Dynamics Models." *Proceedings of the 22nd International Conference of the System Dynamics Society*, Oxford, England.
- Choi, T.Y., K. J. Dooley and M. Rungtusanatham. 2001. "Supply networks and complex adaptive systems: control versus emergence." *Journal of Operations Management* 19 (2001) 351–366.
- Coyle, R.G. 1996. *System Dynamics: A Practical Approach*. Chapman and Hall, London, UK.
- Dangerfield, B. and C. Roberts. 1996. "An Overview of Strategy and Tactics in System Dynamics Optimisation." *Journal of the Operational Research Society*, 47, pp 405-423.
- Duggan, J. 2005. "Using Multiple Objective Optimisation to Generate Policy Insights for System Dynamics Models." *Proceedings of the 23rd International Conference of the System Dynamics Society*, Cambridge, MA.
- Duggan, J. 2006. "Enhancing Policy Analysis in System Dynamics Using Multiple Objective Optimisation". *Proceedings of the International Mediterranean Modelling Multiconference*, edited by A. Bruzzone, A. Guasch, M.A. Piera and J. Rozenblit. ISBN 84-690-0726-2.
- Forrester, J.W. 1961. *Industrial Dynamics*. MIT Press.
- Grossler, A., Stotz, M. and Schieritz, M. 2003. "A Software Interface between System Dynamics and Agent-Based Simulations – Linking Vensim and RePast." *Proceedings of the 21st International Conference of the System Dynamics Society*. New York.
- Grossman, B. 2002. "Policy Optimization in Dynamic Models with Genetic Algorithms" *Proceedings of the 20th International Conference of the System Dynamics Society*, Palermo, Italy.

- Keloharju, R. and E.F. Wolstenholme. 1989. "A Case Study in System Dynamics Optimisation." *J. Ops. Res. Soc.*, Vol. 40, No.3., pp 221-230.
- North, M.J., Collier, N.T., and Vos, J.R. 2006. "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit." *ACM Transactions on Modeling and Computer Simulation*. Vol.16, No.1, pp 1-25.
- Rahmandad, H. 2004. Heterogeneity and network structure in the dynamics of contagion: Comparing agent-based and differential equation models. *Proceedings of 22nd International Conference of the System Dynamics Society*, Oxford, England.
- Schieritz, M and A. Grossler. 2003. "Emergent Structures in Supply Chains— A Study Integrating Agent-Based and System Dynamics Modeling." *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)* 0-7695-1874-5/03
- Scholl, H.J. 2001. "Agent-based and System Dynamics Modeling: A Call for Cross Study and Joint Research." *Proceedings of the 24th Hawaii International Conference on Systems Sciences*, ISSN 0-7695-0981-9/01.
- Sterman, J.D. 1989. "Modeling managerial behaviour: Misperceptions of feedback in a dynamic decision making experiment." *Management Science*. 35 (3), pp 321-339.
- Sterman, J.D. 2000. *Business Dynamics. Systems Thinking and Modeling for a Complex World*. McGraw Hill Higher Education.

A. Appendix 1: Beer Game Optimisation Characterisation

```
<network_optimiser>

<network_model>BeerGameNetworkOPT.xml</network_model>
<number_strategies>6</number_strategies>
<number_nodes>4</number_nodes>
<number_objectives>1</number_objectives>
<optimisation_task>MIN</optimisation_task>
<crossover_rate>0.70</crossover_rate>
<mutation_rate>0.15</mutation_rate>
<number_generations>100</number_generations>
<population_size>100</population_size>

<objectives>

  <objective>
    <id>Objective 1</id>
    <payoff>Global.TotalCosts</payoff>
  </objective>

</objectives>

<strategies>
  <strategy>
    <id>0</id>
    <name>BG_Heuristic_00</name>
    <description>Beer Game Agent - Uses the Supply Line</description>
    <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
    <file>BG_00_TemplateSupplyLine.xml</file>
  </strategy>

  <strategy>
    <id>1</id>
    <name>BG_Heuristic_01</name>
    <description>Beer Game Agent - Does not use the Supply Line</description>
    <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
    <file>BG_01_TemplateNoSupplyLine.xml</file>
  </strategy>

  <strategy>
    <id>2</id>
    <name>BG_Heuristic_02</name>
    <description>Beer Game Agent - Only uses Stock Adjustment</description>
    <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
    <file>BG_02_TemplateASOnly.xml</file>
  </strategy>

  <strategy>
    <id>3</id>
    <name>BG_Heuristic_03</name>
    <description>Beer Game Agent - Only uses expected orders</description>
    <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
    <file>BG_03_TemplateEOOnly.xml</file>
  </strategy>

  <strategy>
    <id>4</id>
    <name>BG_Heuristic_04</name>
    <description>Beer Game Agent - Uses SL and Eexpected Orders</description>
    <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
    <file>BG_04_TemplateSLEO.xml</file>
  </strategy>

  <strategy>
    <id>5</id>
    <name>BG_Heuristic_05</name>
    <description>Beer Game Agent - Only uses the Supply Line</description>
  </strategy>
</strategies>
```

```

                <directory>C:\Program Files\SDPrototype\Models\NETOPT\</directory>
                <file>BG_05_TemplateSLOnly.xml</file>
            </strategy>
</strategies>

<parameters>

<!-- ##### RETAILER ##### -->
<parameter>
<variable>Retailer.SAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Retailer.SLAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Retailer.EOAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
    <strategy>TRUE</strategy>
    <variable>RetailerStrategyID</variable>
    <min_value>0</min_value>
    <max_value>5</max_value>
</parameter>

<!-- ##### WHOLESALE ##### -->
<parameter>
<variable>Wholesaler.SAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Wholesaler.SLAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Wholesaler.EOAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
    <strategy>TRUE</strategy>
    <variable>WholesalerStrategyID</variable>
    <min_value>0</min_value>
    <max_value>5</max_value>
</parameter>

<!-- ##### DISTRIBUTOR ##### -->
<parameter>
<variable>Distributor.SAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>

```

```

<variable>Distributor.SLAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Distributor.EOAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
  <strategy>TRUE</strategy>
  <variable>DistributorStrategyID</variable>
  <min_value>0</min_value>
  <max_value>5</max_value>
</parameter>

<!-- ##### FACTORY ##### -->
<parameter>
<variable>Factory.SAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Factory.SLAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
<variable>Factory.EOAT</variable>
<min_value>1.0</min_value>
<max_value>15.0</max_value>
</parameter>

<parameter>
  <strategy>TRUE</strategy>
  <variable>FactoryStrategyID</variable>
  <min_value>0</min_value>
  <max_value>5</max_value>
</parameter>

</parameters>

</network_optimiser>

```

B. Appendix 2: Beer Game Network Model

```
<network>
  <start_time>0</start_time>
  <finish_time>50</finish_time>
  <time_step>0.25</time_step>

  <nodes>

    <!-- ##### NODE 1 ##### -->
    <node>
      <identifier>Retailer</identifier>
      <number>1</number>
      <template>${STRATEGY_REF_0}</template>

      <number_inputs>1</number_inputs>
      <number_outputs>1</number_outputs>

      <closed_list>
        <variable>${ID$.CustomerDemand}</variable>
      </closed_list>

      <candidates>
        <variable>${ID$.UpstreamShipments}</variable>
        <candidate>
          <node_id>Wholesaler</node_id>
          <variable_id>${ID$.DownstreamShipments}</variable_id>
        </candidate>
      </candidates>

      <inputs>
        <input>Wholesaler</input>
      </inputs>

      <outputs>
        <output>Wholesaler</output>
      </outputs>
    </node>

    <!-- ##### NODE 2 ##### -->
    <node>
      <identifier>Wholesaler</identifier>
      <number>2</number>
      <template>${STRATEGY_REF_1}</template>

      <number_inputs>1</number_inputs>
      <number_outputs>1</number_outputs>

      <inputs>
        <input>Retailer</input>
        <input>Distributor</input>
      </inputs>

      <candidates>
        <variable>${ID$.CustomerDemand}</variable>
        <candidate>
          <node_id>Retailer</node_id>
          <variable_id>${ID$.OutgoingOrders}</variable_id>
        </candidate>
      </candidates>

      <candidates>
        <variable>${ID$.UpstreamShipments}</variable>
        <candidate>
          <node_id>Distributor</node_id>
          <variable_id>${ID$.DownstreamShipments}</variable_id>
        </candidate>
      </candidates>
  </nodes>
</network>
```

```

        <outputs>
            <output>Retailer</output>
            <output>Distributor</output>
        </outputs>
    </node>

<!-- ##### NODE 3 ##### -->
<node>
    <identifier>Distributor</identifier>
    <number>3</number>
    <template>${STRATEGY_REF_2}</template>

    <number_inputs>1</number_inputs>
    <number_outputs>1</number_outputs>

    <inputs>
        <input>Factory</input>
        <input>Wholesaler</input>
    </inputs>

    <candidates>
        <variable>${ID$.CustomerDemand}</variable>
        <candidate>
            <node_id>Wholesaler</node_id>
            <variable_id>${ID$.OutgoingOrders}</variable_id>
        </candidate>
    </candidates>

    <candidates>
        <variable>${ID$.UpstreamShipments}</variable>
        <candidate>
            <node_id>Factory</node_id>
            <variable_id>${ID$.DownstreamShipments}</variable_id>
        </candidate>
    </candidates>

    <outputs>
        <output>Factory</output>
        <output>Wholesaler</output>
    </outputs>
</node>

<!-- ##### NODE 4 ##### -->
<node>
    <identifier>Factory</identifier>
    <number>4</number>
    <template>${STRATEGY_REF_3}</template>

    <number_inputs>1</number_inputs>
    <number_outputs>1</number_outputs>

    <closed_list>
        <variable>${ID$.UpstreamShipments}</variable>
    </closed_list>

    <candidates>
        <variable>${ID$.CustomerDemand}</variable>
        <candidate>
            <node_id>Distributor</node_id>
            <variable_id>${ID$.OutgoingOrders}</variable_id>
        </candidate>
    </candidates>

    <inputs>
        <input>Distributor</input>
    </inputs>

    <outputs>
        <output>Distributor</output>

```

```

        </outputs>
    </node>

</nodes>

<aggregates>
    <aggregate>
        <auxiliary>
            <name>Global.TotalCosts</name>

            <equation>Retailer.TotalCosts+Wholesaler.TotalCosts+Distributor.TotalCosts+Factory
.TotalCosts</equation>
        </auxiliary>
    </aggregate>

    <aggregate>
        <auxiliary>
            <name>Global.InventoryCost</name>
            <equation>0.5</equation>
        </auxiliary>
    </aggregate>

    <aggregate>
        <auxiliary>
            <name>Global.StockoutCost</name>
            <equation>2.0</equation>
        </auxiliary>
    </aggregate>

</aggregates>

</network>

```

C. Appendix 3: Beer Game Sample Strategy File

```
<template>
  <interfaces>
    <interface>
      <var>$ID$.CustomerDemand</var>
      <type>write</type>
      <linked_to>$ID$.OutgoingOrders</linked_to>
      <closed_value>100+STEP(100,5)</closed_value>
    </interface>

    <interface>
      <var>$ID$.UpstreamShipments</var>
      <type>write</type>
      <linked_to>$ID$.DownstreamShipments</linked_to>
      <closed_value>$ID$.OutgoingOrders</closed_value>
    </interface>

    <interface>
      <var>$ID$.DownstreamShipments</var>
      <type>read</type>
    </interface>

    <interface>
      <var>$ID$.OutgoingOrders</var>
      <type>read</type>
    </interface>
  </interfaces>

  <stocks>
    <stock>
      <name>$ID$.ExpectedOrders</name>
      <init>100</init>
      <inflow>$ID$.CEO</inflow>
    </stock>

    <stock>
      <name>$ID$.OrderQueue</name>
      <init>100</init>
      <inflow>$ID$.CustomerDemand</inflow>
      <outflow>$ID$.OrdersFulfilled</outflow>
    </stock>

    <stock>
      <name>$ID$.Stock</name>
      <init>400</init>
      <inflow>$ID$.Arrivals</inflow>
      <outflow>$ID$.DownstreamShipments</outflow>
    </stock>

    <stock>
      <name>$ID$.SupplyLine</name>
      <init>300</init>
      <inflow>$ID$.UpstreamShipments</inflow>
      <outflow>$ID$.Arrivals</outflow>
    </stock>

    <stock>
      <name>$ID$.TotalCosts</name>
      <inflow>$ID$.ChangeTC</inflow>
      <init>0.0</init>
    </stock>
  </stocks>

  <flows>
```

```

<flow>
  <name>$ID$.CustomerDemand</name>
  <equation>100</equation>
</flow>

<flow>
  <name>$ID$.CEO</name>
  <equation>$ID$.Error/$ID$.EOAT</equation>
</flow>

<flow>
  <name>$ID$.OrdersFulfilled</name>
  <equation>MIN($ID$.OrderQueue,$ID$.Stock)</equation>
</flow>

<flow>
  <name>$ID$.Arrivals</name>
  <equation>DELAYFIXED($ID$.UpstreamShipments,3,100)</equation>
</flow>

<flow>
  <name>$ID$.DownstreamShipments</name>
  <equation>$ID$.OrdersFulfilled</equation>
</flow>

<flow>
  <name>$ID$.UpstreamShipments</name>
  <equation>$ID$.OutgoingOrders</equation>
</flow>

<flow>
  <name>$ID$.ChangeTC</name>
  <equation>$ID$.Cost</equation>
</flow>
</flows>

<auxiliaries>
  <auxiliary>
    <name>$ID$.SupplyLineAdjustment</name>
    <equation>($ID$.DesiredSupplyLine-$ID$.SupplyLine)/$ID$.SLAT</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.DesiredStock</name>
    <equation>400</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.DesiredSupplyLine</name>
    <equation>$ID$.ExpectedOrders*3</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.Error</name>
    <equation>$ID$.CustomerDemand-$ID$.ExpectedOrders</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.EOAT</name>
    <equation>3</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.NetStock</name>
    <equation>$ID$.Stock-$ID$.OrderQueue</equation>
  </auxiliary>

  <auxiliary>
    <name>$ID$.OutgoingOrders</name>

```

```

    <equation>MAX(0,$ID$.ExpectedOrders+$ID$.StockAdjustment+$ID$.SupplyLineAdjustment
) </equation>
</auxiliary>

<auxiliary>
    <name>$ID$.SAT</name>
    <equation>1</equation>
</auxiliary>

<auxiliary>
    <name>$ID$.SLAT</name>
    <equation>1</equation>
</auxiliary>

<auxiliary>
    <name>$ID$.StockAdjustment</name>
    <equation>($ID$.DesiredStock-$ID$.Stock)/$ID$.SAT</equation>
</auxiliary>

<auxiliary>
    <name>$ID$.InventoryStatus</name>
    <equation>IS_NEGATIVE($ID$.NetStock)</equation>
</auxiliary>

<auxiliary>
    <name>$ID$.Cost</name>
    <equation>ABS($ID$.NetStock) * (Global.InventoryCost+(Global.StockoutCost-
Global.InventoryCost)*$ID$.InventoryStatus)</equation>
</auxiliary>

</auxiliaries>
</template>

```