Computational Phenomena Occurring Solving the Binary Quadratic Assignment Problem

Ian Davidson
Division of Building, Construction and Engineering
Commonwealth Scientific and Industrial Research Organisation, Australia
email: inpd@mel.dbce.csiro.au

Abstract

Computational phenomena occurring whilst solving the binary quadratic assignment problem using simulated annealing is investigated. The presence of phenomena such as self organisation and emergent computation are searched for in both the optimisation process and the solutions. Implicit emergent computation is shown to be facilitated by a simple perturbation operator. The potential for the application of the research is discussed

1. Introduction

Stochastic combinatorial optimisation techniques such as simulated annealing, have been very successful at solving a variety of combinatorial optimisation problems. They have a quality of robustness which enables them to produce good solutions to many different problems.[1] However, with large real world problems, the time required for the simulated annealing technique to converge is great.[2] Simulated annealing has been studied indepth as a Markov chain.[3] Greater insight into what is computationally occurring may perhaps offer an answer to improving the quality of the solution, whilst reducing the convergence time. In this study, computational phenomena occurring solving the binary quadratic assignment problem using simulated annealing is investigated.

2. Why Should Computational Phenomena Occur ?

It is well known that simulated annealing is a successful combinatorial optimisation technique. It is successful because if the simulated annealing process is kept at thermal equilibrium, the distribution of configurations with respective to their energy levels (costs) is given by the Boltzmann distribution. If the system reaches equilibrium at each and every temperature given by the annealing schedule then the distribution of configurations will converge to the theoretical ground state(s).[3] In simulated annealing, these ground states correspond to the optimal configuration.

Simulated annealing is modelled on the annealing of solids. The actual annealing process can be explained from techniques in condensed matter physics. During the annealing of solids, the material goes through a phase transition which has been shown to occur in the simulated annealing process. Kirkpatrick in his studies on simulated annealing showed that the behaviour of specific heat of the system as a function of temperature indicated the presence of a phase transition.[4]

Von Neumann proved that cellular automata's (CA) were capable of universal computation through his creation of a self-reproducing machine.[5] Wolfram in [6] divided the macroscopic behaviour of CA's into four classes, analogies with dynamic systems are provided in parentheses:

Class I:	Converges to homogenous state (limit points)
Class II:	Converges to separate periodic structures (limit cycles)
Class III:	Chaotic aperiodic patterns (strange attractor)
Class IV:	Complex patterns of localised structures (possess long transients)

Wolfram postulates that due to their localised structures and long transients CA's in class IV are capable of supporting the mechanisms which can sustain universal computation.

Langton in his studies of CA's [7] suggested that CA's in a class IV behaviour undergo phase transitions, especially second order phase transitions. He concludes that the presence of these phase transitions and give rise to the possibility of emergent computation occurring. Cellular automata and simulated annealing can be both viewed as dynamical systems sharing the property of both being Markov processes. Their behaviour can be explained by the same field (statistical mechanics) and they both share common phenomena (phase transitions). Given this parallel, it is investigated that if the simulated annealing system is at equilibrium, does emergent computation occur. If the system is indeed at thermodynamic equilibrium and phase transitions occur, does this necessitate the emergence of computations that will give rise to better solutions of the problem ?

3. The Binary Quadratic Assignment Problem

The binary quadratic assignment problem consists of placing *N* activities into *M* zones. The zones are spatially apart with a given distance between zones. Activities interact with each other and there is specified cost per unit distance. The objective is to minimise the total cost of interaction. The problem and objective can be expressed as:

Let	N	= the number of activities
	M	= the number of zones
	f_{ik}	= the interaction between activity i and k
	d_{jl}	= the distance between zone j and zone l
	b_{ijlk}	= the cost per unit of interaction from activity i in zone j to activity k in zone l

The objective function is:

Minimise

$$\sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{k=1}^{N} \sum_{l=1}^{M} b_{ijkl} . f_{ik} . d_{jl}$$

This type of problem occurs extensively in the real world in many forms. The problem is classified as a NP hard problem. In this paper we shall discuss the facilities layout problem which has the additional constraints that:

> N = M
> The size of N equals the Size of M, that is an activity occupies one and only one zone.

For the purpose of problem expression, activities are aggregated into activity types.

4. Optimisation Technique

The fundamental optimisation technique used for this study is detailed below.

4.1 Fundamental Algorithm for Maximisation

```
1.  Choose an initial configuration
2.  Optimise:  Choose a new configuration  which is a small perturbation from the old
               configuration
3.             If quality of new configuration > quality of old configuration
4.                 old configuration := new configuration
5.             Else with probability $\log_e(\Delta Cost/CurrentTemperature)$ accept
6.                 old configuration := new configuration
7.             If after a prescribed time no increase in quality of solution, reduce
                   temperature according to annealing schedule
8.             If after a long time no increase in quality is found or too many iterations have
                   occurred Stop
9.             Goto Optimise
```

The initial configuration is randomly generated, given the lack of constraints in this type of problem an initial configuration is easily generated. Stochastic optimisation techniques such as simulated annealing start with an initial configuration and iteratively slightly perturbate the previous configuration. The problem representation specifies the dimensionality of the space to explore. The perturbation operator and the optimisation technique determines the manner it will be explored in.

The perturbation operator used in this study is an exchange of activities contained in two randomly selected zones.

5. Emergent Computation

Forrest in [8] states that emergent global behaviour can arise from many local interactions. However when the emergent behaviour is also a measurable computation, we can view this behaviour as being an emergent computation. Most studies on emergent computation have been on CA's, and in their context, emergent computation is supported by exploiting primitive components. There exists no analogous primitive components in simulated annealing to the primitive components in cellular automata. Rather, the primitive component is a process, the perturbation operator. Therefore, generally we can state that exploiting explicit microscopic behaviour gives rise to emergent macroscopic behaviour.

The constituents of emergent computation are given in [8]. In relation to this study they can be redefined as:

- A collection of explicit process[es] (the perturbation operator(s)) applied to search the configuration space.
- A series of explicit process[es] (the perturbation operator(s)) together form implicit processes.
- A natural interpretation of the macroscopic behaviour of the processes as being a computation not explicitly capable of being achieved by a single application of an explicit process.

5.1 An Example of Emergent Computation

The perturbation operator in this study is a simple activity swap operator.

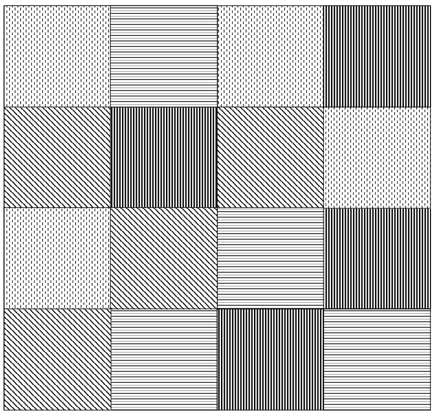
Figure 1: Configuration at Iteration 1

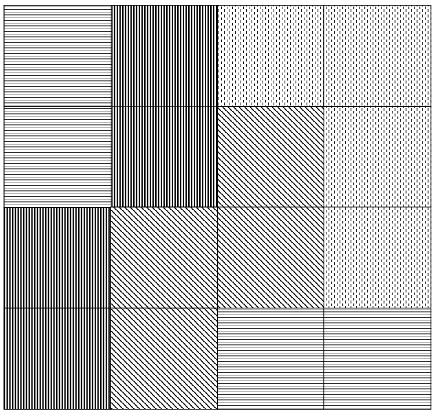
Figure 2: Configuration at Iteration 100

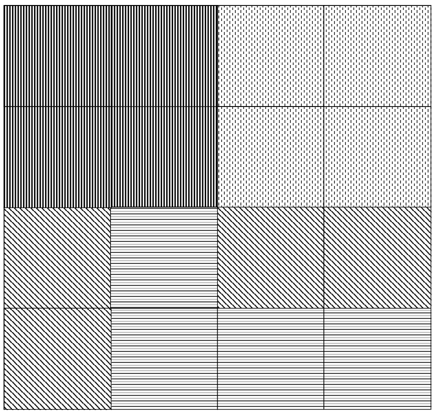
Figure 3: Configuration at Iteration 5000

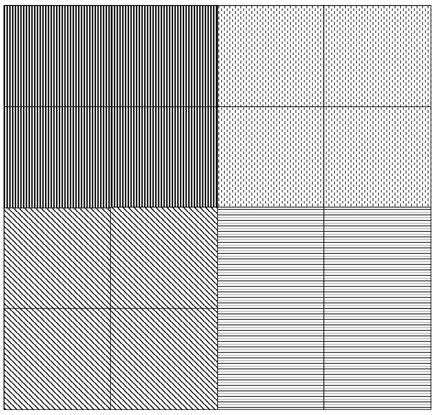
Figure 4 Optimum Configuration

For a simple trial problem of:

N=16 (Four activity types of four activities each)
M=16

The interaction between the activity types is weak (inter activity forces) but strong between activities of the same type (intra activity forces). The initial configuration is shown in figure 1. The optimum configuration is shown in figure 4. The optimum configuration is never reached, even after 1 million iterations using the single activity perturbation operator. Under this circumstance the most chosen alternative is to continually restart the entire optimisation process with new initial configurations.

As the figure 2 clearly shows at a very early stage of the run similar activity types aggregate in pairs. The remainder of the run in most cases only accepts a sequence of swaps that result in an entire pair being moved to a new location. That is, once a pair is formed the intra activity forces are too great, hence it is never is broken (figure 3). Clearly whilst the explicit process is to swap two activities, after iteration 100 the optimisation process implicitly is only accepting swaps that result in paired activities being swapped. If the annealing schedule was too quick and "quenching" occurred the aggregation into pairs occurred only in approximately 25% of the configuration. The annealing schedule is primarily determined by the cooling rate *C* where *C* is a real number between 0 and 1. The temperature at time T_t is given by $T_{t-1}.C$. If *C* is below 0.925 then the amount of pair aggregation reduces considerably and the emergent perturbation operators cannot occur.

The outlined example is simple for clarity. Real world size problem consists of 2000 zones and 45 activity types, each activity type contains between 1 to 15 individual activities. Similar phenomena of clustering and problems moving clusters are found in such problems, especially given the clusters are not limited to being square but can be linear, "circles", rectangular and even in extreme cases, hexagons. However with the collection of perturbation operators discussed in the next section, better quality solutions were reached.

6. Uses of Emergent Computation

Clearly a use of the illustrated emergent computation is to make the emergent behaviour explicit. That is, add another perturbation operator that allows the exchange of two sets of two adjacent activities. This technique was used to obtain the optimum configuration (figure 4) by initially applying the original single activity swap perturbation operator and later the new pair activity swap perturbation operator.

However the greater potential for emergent computation can be reached if a series of perturbation operators can be used together in the same optimisation run. Such a situation would facilitate a richer combination of emergent perturbation operators that could reduce the optimisation time and improve the quality of the solution. This would be a powerful technique as it would provide the optimisation technique with a flexible approach to handling different problem types. Every optimisation problem type (such as the BQAP) has its own specific configuration space that the optimisation technique searches. Configuration spaces are dependant on the problem representation and may have unique qualities such as being highly constrained that could make movement between feasible solutions difficult. Often it is a time consuming process to explicitly code the perturbation operator to efficiently search the solution space. By having a combination of available explicit perturbation operators that this could give rise to additional emergent computation, this would remove this task from the implementor.

The following perturbation operators have been trialed on optimisation runs of binary quadratic assignment problems of various size. The perturbation operators are designed to introduce a small perturbation to the existing configuration.

Horizontal Line Swap

Swaps a horizontal line of activities of length *n* where *n* is a random number between 1 and *N/(N/8)*

Vertical Line Swap

Swaps a vertical line of activities of length *n* where *n* is a random number between 1 and *N/(N/8)*

Diagonal Line Swap

Swaps a diagonal line of activities of length *n* where *n* is a random number between 1 and *N/(N/16)*

Block Swap

Swaps a block of activities of length and width *n* where *n* is a random number between 1 and *N/(N/16)*

Circular Swap

Selects a random activity and a random radius *r* where *r* is a random number between 1 and *N/(N/16)*. Progressively swaps all activities in the neighbourhood given by *r* with the centre activity until a cost decrease is established or the neighbourhood is exhausted.

A perturbation operator was randomly selected and applied each and every iteration. The majority of trialed optimisation runs resulted in an improvement in the quality of solution than for runs using the single swap perturbation operator. The random selection of perturbation operator is the initial and obvious choice on coordinating multiple perturbation operator. The author hopes to implement a genetic algorithm to evolve the optimum sequence of perturbation operators with respect to problem type and simulated annealing control variables.

These operators together have been trialed and are capable of handling many difficult problems where clustering of activity types has occurred. Clustering may be in number of patterns such as blocks, diamonds, lines and others. It is believed that the these perturbation operators can handle the majority of these patterns.

7. Conclusion

The author feels that greater insight into what is computionally occurring is a valid method of attempting to reduce the time and improve the quality of the optimisation process. Langton has discussed the notion of universal computation being supported in CA's if in Wolfram's Class 4 behaviour where phase transitions occur. Simulated annealing with the correct annealing schedule keeps the system at thermodynamic equilibrium. Kirkpatrick illustrated that simulated annealing undergoes phase transitions as a actual solid annealing does.

Hence it is feasible that given both CA's and simulated annealing can be viewed as dynamical systems there exists the possibility of some sort of implicit computation occurring during a simulated annealing run. It is shown that there exists implicit or emergent computation occurring in one form of the binary quadratic assignment problem if the annealing schedule is slow enough. An explicit single swap perturbation operator gives rise to an implicit pair swap perturbation operator. The possibility of using a combination of explicit perturbation operators that give rise to implicit emergent perturbation operators is discussed. This approach was used on large real world problems with promising results. Such an approach can potentially offer a powerful and flexible method of reducing the time to implement the optimisation technique (there is no need to specifically trial and test perturbation operators), reducing the optimisation time and can often find a better solution that was incapable of being found with only one perturbation operator.

8. References

[1] Dueck G., Scheuer T (1990), "Threshold Accepting: A General Purpose Optimisation Algorithm Appearing Superior to Simulated Annealing", *Journal of Computational Physics,* **90**, pp 161-175.

[2] Geman S, Geman D (1984), "Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images", *IEEE Transactions On Pattern Analysis and Machine Intelligence,* **Vol. PAMI6 No 6**.

[3] van Laarhoven P.J.M., Aarts E.H.L (1987), *Simulated Annealing: Theory and Applications,* D. Reidel Publishing Company.

[4] Kirkpatrick S., Gelatt C.D, Vecchi M.P (1982) , *IBM Research Report RC 9355.*

[5] J. von Neumann (1966), *Theory of Self-Reproducing Automata,* , University of Illinois Lectures on the Theory and Organisation of Complicated Automata, ed. A.W. Burks, University of Illinois Press, Urbana IL.

[6] S. Wolfram (1984), "Universality and complexity in cellular automata", *Physica D 10, pp* 1- 35.

[7]C. G. Langton (1990), "Computation At The Edge of Chaos: Phase Transitions and Emergent Computation", *Physica D 42*, pp 12-37.

[8] S. Forrest editor (1990), *Emergent Computation*, MIT/North-Holland, pp 1-11