Phase-Transitions: Efficient Algorithms and Combinatorial Properties (An Overview)

Paul E. Dunne and Michele Zito
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF
Great Britain

{ped,michele}@csc.liv.ac.uk

Forsan et haec olim meminisse iuvabit

Phase-Transitions and Thresholds

Informal definition:

A ‘*small*’ change in some *parameter* of a ‘system’ produces a ‘*large*’ effect on some *property* of the same system.

The parameter value determining when the effect occurs can be interpreted as a ‘critical’ *threshold*.

These phenomena occur and have been studied in a very diverse variety of settings.

For example:

State changes (e.g. solid-liquid) through ‘temperature’ variation.

Modelling the spread of epidemics.

Reliability of communication networks.

Ferromagnetic models

Evolutionary graph theory

Behaviour of search processes

Efficiency of algorithms.

The modelling of these phenomena has evolved within disciplines such as:

Percolation Theory

A ‘system’ is a (infinite) lattice of ‘sites’. Adjacent sites are connected with some probability p. Connectivity properties are affected as p is varied. (1957+, Broadbent)

Statistical Mechanics

Using ‘effective field’ models (e.g. Ising, Potts, Gibbs). (1800s: Laplace)

Random Graph Theory

The ‘system’ is an n-vertex graph; edge included with probability p (that may depend on n). First studied in context of probability of connectivity w.r.t. $p(n)$. (1955+, Erdős)

Computer Science

von Neumann (1956) examined problems in reliable hardware design

An ε-noise network is a network of (fixed arity) logic gates, each of which fails independently with probability ε.

For ε 'small enough', there exists $p > 0.5$, such that for any n-argument logic function, f, there is an ε-noise network C_f, whose computation on an any given α, equals $f(\alpha)$ with probability at least p.

Much of the Computer Science interest is, however, comparatively recent, and has, in part, derived from experimental studies in **Artificial Intelligence**:

Experimental study of average time taken by an ordered search process on randomly chosen inputs. (1992+, Kirkpatrick *et al.*)

The class of computational problems particularly of interest, are often defined in terms of:

a) A set of problem *instances* I

b) a set of *candidate solutions* W

c) A binary relation $R \subseteq I \times W$

The two computational problems that have been the object of algorithmic study are:

1) The *decision problem* for R: given $x \in I$ $\exists ? w \in W \ (x, w) \in R$.

2) The *search problem* for R: given $x \in I$ find $w \in W$ s.t. $(x, w) \in R$.

For example:

3–*SAT*: $I =$ n-argument Boolean formulae in 3-*CNF*; $W = \bigcup_n \langle 0,1 \rangle^n$; $R = (F, \alpha)$ where $F(\alpha) = 1$.

HAM: $I =$ n-vertex undirected graphs; $W =$ n-vertex *cycles*; $R = (G, C)$ where C is a sub-graph of G.

Computational Complexity Theory defines the class of *efficiently* computable problems to be those for which:

There is an algorithm that halts in at most n^k steps on all instances encoded in n bits, where $k \in \mathbf{N}$.

The set of decision problems in this class is denoted P (polynomial-time)

A number of classical decision problems are not known to be in P.

Following work of Steven Cook in 1971, much attention has focused on the relationshop between the sets P and NP, the latter being the set of relations, R, for which $(x, w) \in ?\, R$ can be decided efficiently.

Phase-Transitions and Decision Problems

The number of candidate solutions to test for an instance, x is often exponentially large in terms of the number of bits encoding x:
e.g. 3–*SAT*: F can be encoded in $4n(n-1)(n-2)/3$ bits; 2^n candidates, α.
HAM: G can be encoded in $n(n-1)/2$ bits; $(n-1)!/2$ candidate cycles.

Both decisions problems are in *NP*, however, it is strongly conjectured that neither is in *P*: The only (known) methods require testing all possible candidates for a *at least one* instance.
For instances within certain classes, it may be possible to 'do better on average'.

Suppose X_s is the the of s-bit strings and $X_{s,t}$ the subset of these containing exactly t 1s.

If $s = 4n(n-1)(n-2)/3$ (as in 3-SAT): then $F \in X_{s,t}$,
$3-SAT(F) = 1$ if $t < 8$; $3-SAT(F) = 0$ if $t > 7n(n-1)(n-2)/6$

If $s = n(n-1)/2$ (as in HAM): then $G \in X_{s,t}$
$HAM(G) = 0$ if $t < n$; $HAM(G) = 1$ if $t > n^2/4$

Suppose we consider selecting $x \in X_s$, by choosing each bit at random, according to some probability $p = p(s)$.

Then, one can define, for a decision problem $R:X_s \rightarrow \{0,1\}$, (as above)

$$\mu_p(R) = P[R(x)=1\}$$

Similarly, if A_D and A_S are algorithms solving the *decision* (resp. *search*) versions of *R*, one can define the expected (average) time to decide (resp. search) under the implied distribution.

For many decision problems, *R*, in *NP*, a *critical probability*, $p_c(s)$ is known to exist, which has the following property:

$$\begin{array}{ll} \lim_{s \rightarrow \infty} \mu_p(R)=0 & \text{if } p < p_c(s) \\ \lim_{s \rightarrow \infty} \mu_p(R)=1 & \text{if } p > p_c(s) \end{array} \qquad (\Theta)$$

Several algorithmic questions are motivated by this:

Suppose R is a problem for which a critical probability, as in (Θ), exists:

a) A method which counts the number of 1s in an instance and decides the result on this basis, will *almost always* give correct answers. Is it possible to improve upon this very crude approach?

a1) by using the properties of 'almost all' instances to construct a decision algorithm with efficient *expected* run-time that is *always* correct.

a2) In applications, usually the search problem has to be solved: can properties of 'almost all' instances be exploited in constructing efficient (on average) search algorithms?

b) Experimental studies of search templates suggests that (e.g. in 3–*SAT*), random instances ‘away from’ the critical threshold can be resolved ‘quickly’, but that the search time peaks in the neighbourhood of the critical region. Can an analytic quantification of the variation in average run-time be constructed?