Using the renormalization group to classify Boolean functions

S. N. Coppersmith
Department of Physics
University of Wisconsin, Madison

J. Stat. Phys. **130**, 1065 (2008)

Outline

- Boolean functions: review
- Why classify Boolean functions?
- Renormalization group (RG): review
- RG transformation for Boolean functions and definition of phases
  - Generic phase
  - Non-generic functions and non-generic phases
- Possible relations between phases and computational complexity classes

Boolean functions

- Boolean variable takes on one of two values (1 or 0)
  (= Ising spin)
- A Boolean function is a function of Boolean inputs that yields a Boolean output (can be defined by specifying output value for each of the 2^N input configurations)
- There are 2^{2^N} Boolean functions of N variables

A Boolean function of 3 Boolean variables:

x_1	x_2	x_3	$f(x_1,x_2,x_3)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Classifying Boolean functions gives insight into difference between "typical" and "atypical" Boolean functions

A "typical" Boolean function has output values chosen to be either zero or one independently and randomly for each input configuration

Almost all Boolean functions are "typical"

But almost all functions cannot be feasibly computed (e.g., in time that grows at most polynomially with N).

# of Boolean functions of N Boolean arguments

$$2^{2^N} >> 2^{AN^B}$$

# of Boolean functions that can be computed with resources bounded polynomially with N

C.Shannon, Bell System Technical Journal **28**, 59-98 (1949)

How to tell if an individual realization is “generic”?

(Note that can convert functions to strings by listing outputs for input configurations in lexigraphical order)

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0	1	0	0	1	1	0	1	0	0	1	1	1	0	0	1

These two sequences are equally likely to be found by selecting values independently and randomly to be 0 or 1 with equal probability.

But the second sequence is more “typical.”

How does one determine whether or not a function is typical?

This is a key question in the field of computational complexity, the study of how computational resources needed to solve a problem grow with size of problem specification. (One way to show a function is not typical is to display an algorithm for computing it efficiently.)

This talk: Renormalization group (RG) approach to classifying Boolean functions.

Renormalization group transformations and phase transitions in condensed matter systems

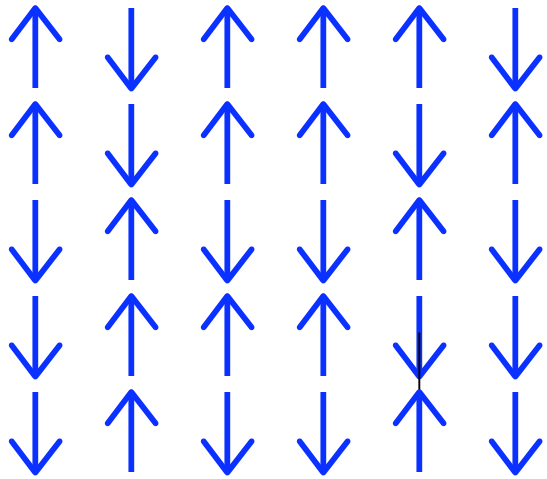
L. Kadanoff, 1966; K. Wilson, 1970

example: magnet

Ferromagnet: spins aligned

Paramagnet: spins random

RG: eliminate spins, creating "effective interactions" between remaining spins



Renormalization group transformations and phase transitions in condensed matter systems

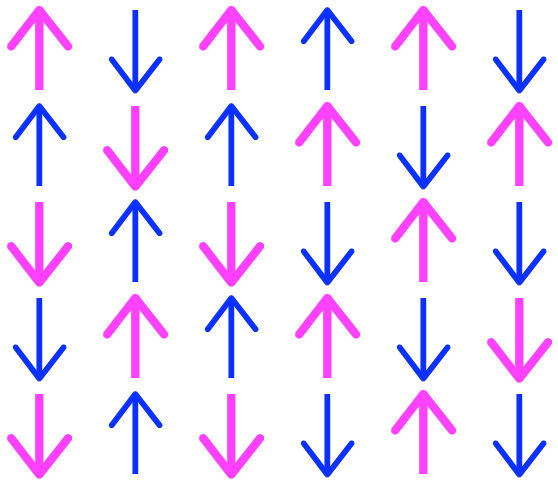
L. Kadanoff, 1966; K. Wilson, 1970

example: magnet

Ferromagnet: spins aligned

Paramagnet: spins random

RG: eliminate spins, creating "effective interactions" between remaining spins



Renormalization group transformations and phase transitions in condensed matter systems

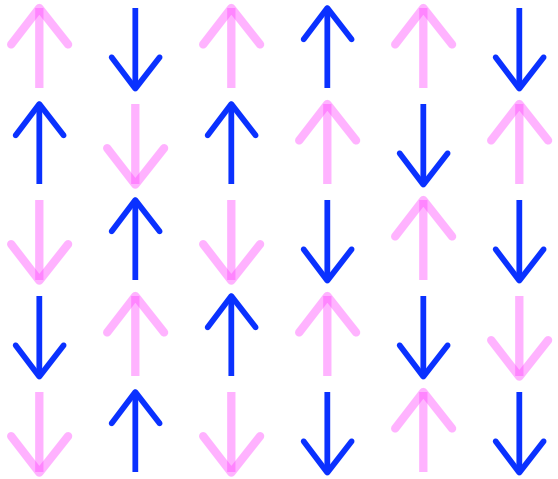
L. Kadanoff, 1966; K. Wilson, 1970

example: magnet

Ferromagnet: spins aligned

Paramagnet: spins random

RG: eliminate spins, creating "effective interactions" between remaining spins



Renormalization group transformations and phase transitions in condensed matter systems

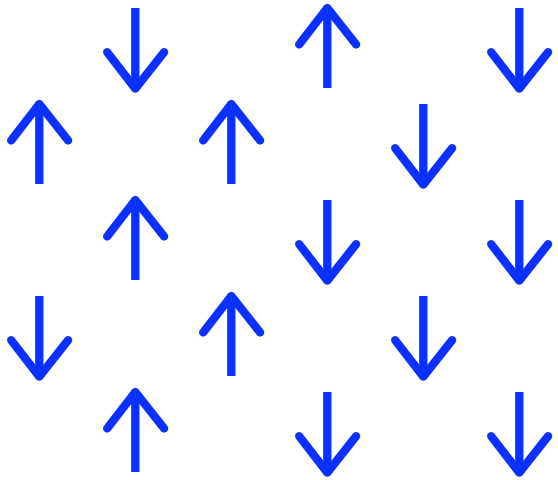
L. Kadanoff, 1966; K. Wilson, 1970

example: magnet

Ferromagnet: spins aligned

Paramagnet: spins random

RG: eliminate spins, creating new configuration with (possibly) different “effective interactions”



Renormalization group transformations and phase transitions in condensed matter systems

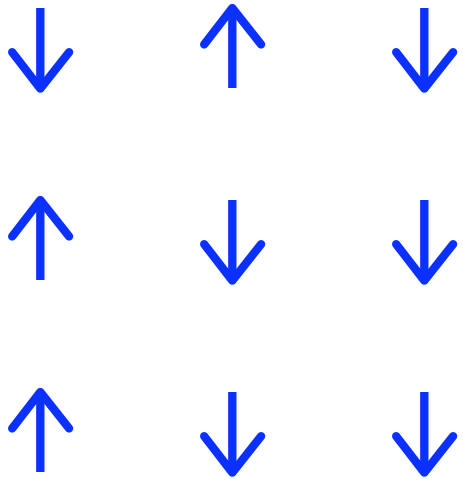
L. Kadanoff, 1966; K. Wilson, 1970

example: magnet

Ferromagnet: spins aligned

Paramagnet: spins random

RG: eliminate spins, creating new configuration with (possibly) different “effective interactions”



Recall RG transformation applied to individual configurations of Ising model for a magnet

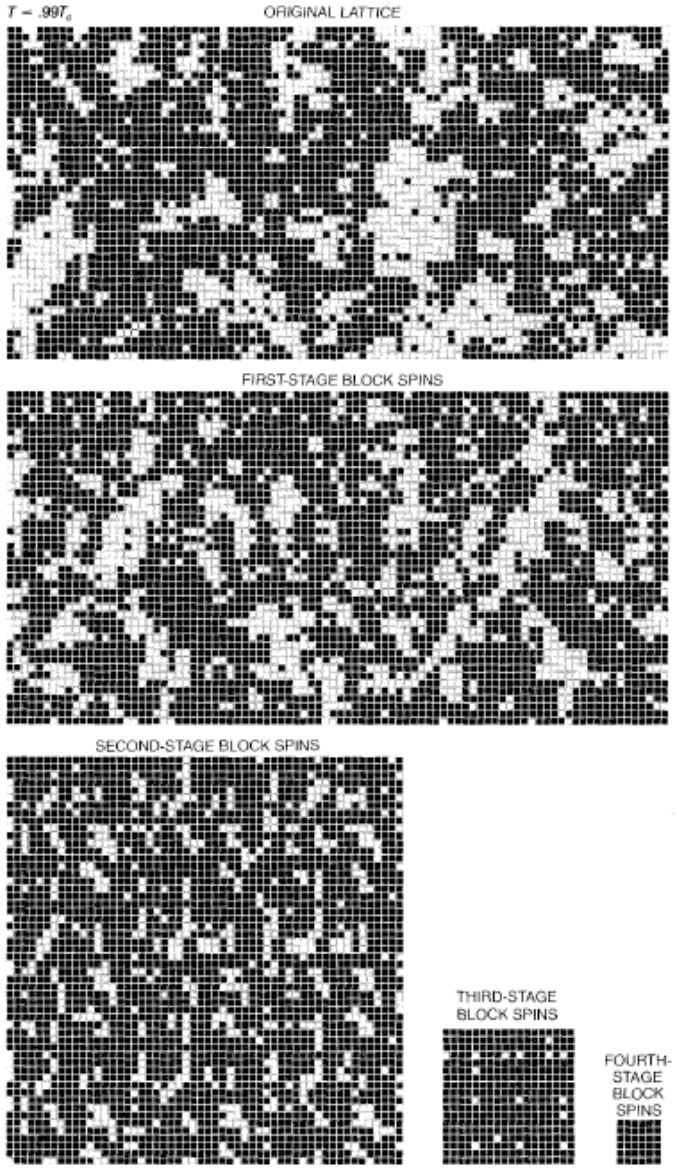
K.G. Wilson, Scientific American 241(2), 158 (1979)

$T > T_c$ (paramagnet)

flow is to disordered configuration

$T < T_c$ (ferromagnet)

flow is to configuration with all spins aligned



replace 3x3 block by single spin

Using the renormalization group to classify Boolean functions

- Recall that renormalization group transformations eliminate degrees of freedom

- A transformation that can be applied to any Boolean function $f(x_1,x_2,\ldots,x_N)$:

$$f(x_1,x_2,\ldots,x_N) \Rightarrow f(0,x_2,\ldots,x_N) \oplus f(1,x_2,\ldots,x_N)$$

transforms function of N variables into one of N-1 variables

$\oplus$ = addition modulo 2

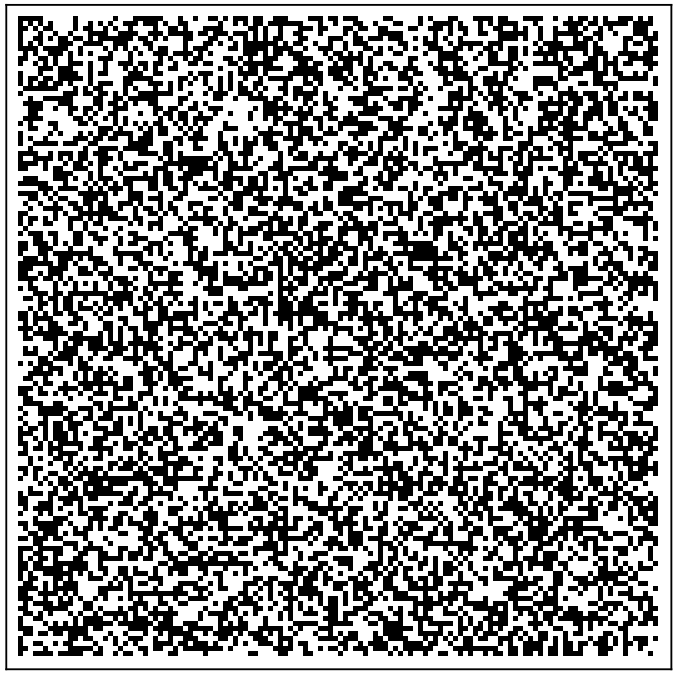
RG transformation yields "fixed point" behavior for generic Boolean functions.

$$f(x_1,x_2,\ldots,x_N) \Rightarrow f(0,x_2,\ldots,x_N) \oplus f(1,x_2,\ldots,x_N)$$

- Generic Boolean function: for each input configuration, the output is chosen independently and randomly to be either 1 or 0 with equal probability
- Apply RG: each output of every resulting function is independently and randomly chosen to be 1 or 0
- RG yields a fixed point $\Rightarrow$ there is a "generic phase"

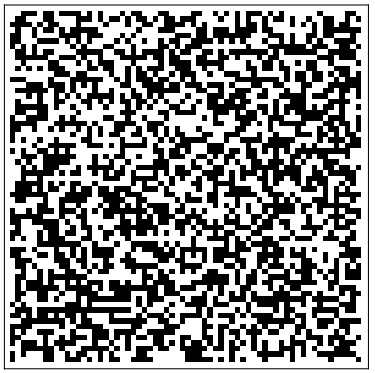
RG maps “typical” Boolean function to another “typical” Boolean function: fixed point behavior

“typical” function -- output for each input chosen independently & randomly to be zero or one with equal probability



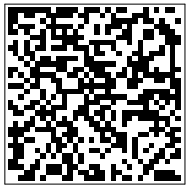
original function

N=14



after 2 renormalizations

N=12



after 4 renormalizations

N=10

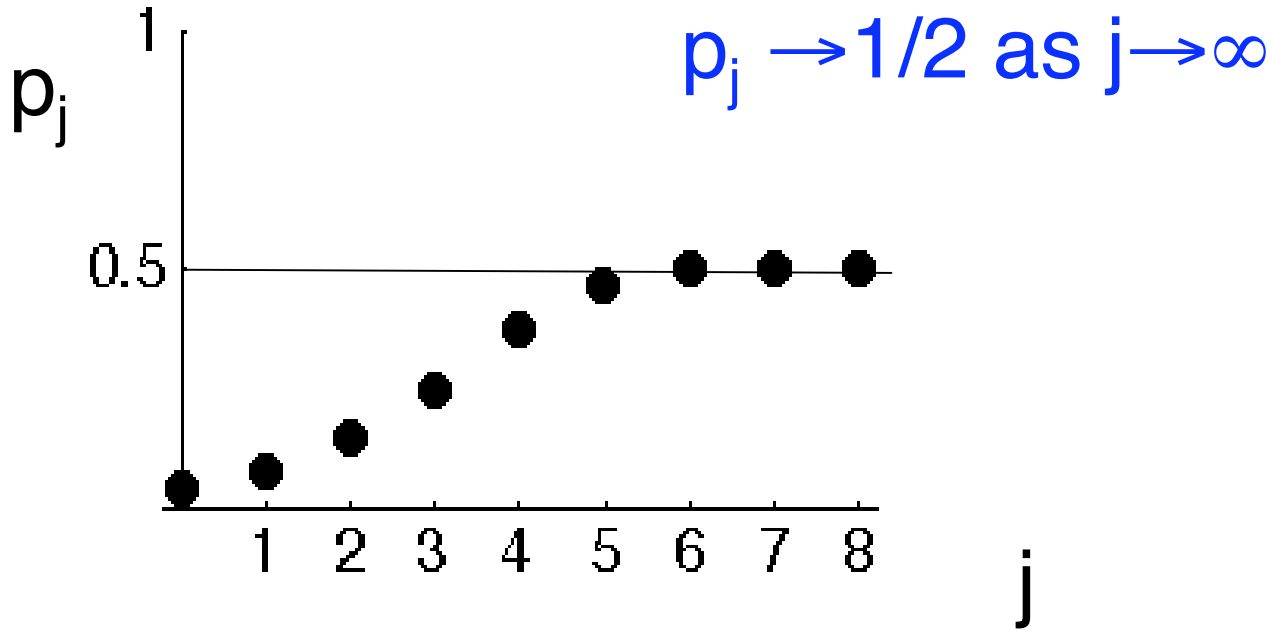
after 6 renormalizations

N=8

RG “flow” in “generic phase”

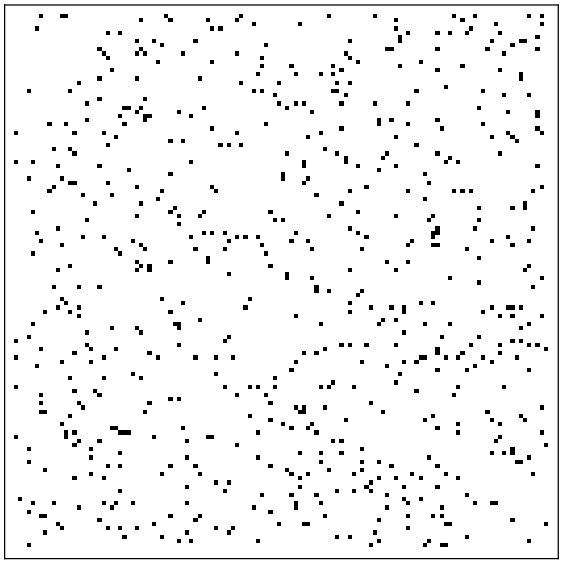
Start with function in which output value for given input is 1 with probability p_0, and values are chosen independently and randomly.

$\Rightarrow p_{j+1}=2p_j(1-p_j)$ (p_j=value of p after j applications of RG)



“Flow” to generic fixed point upon application of the RG

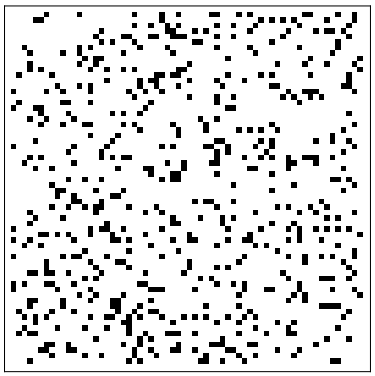
e.g., function where output values are independently and randomly chosen to be one with probability p=0.04



N=14

prob (black)=0.04

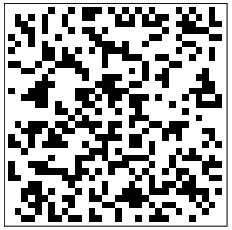
frac(black)=0.0402



after 2 renormalizations

N=12

frac(black)=0.141



after 4 renormalizations

N=10

frac(black)=0.388

after 6 renormalizations

N=8

frac(black)=0.473

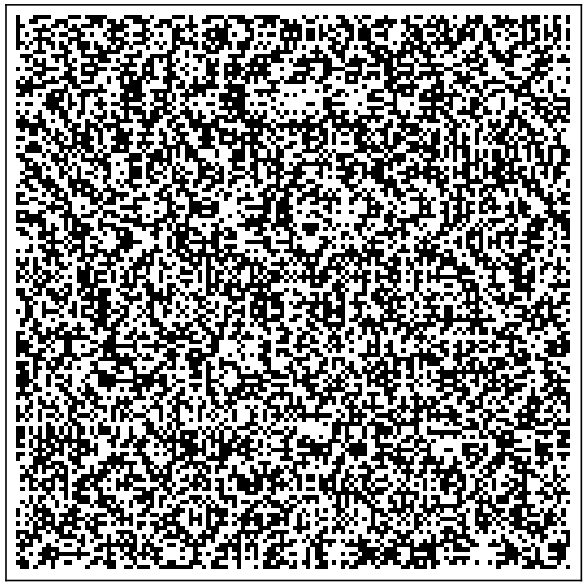
Some functions are non-generic (as seen by behavior upon renormalization)

- Polynomials of order ξ: RG yields zero after ξ+1 iterations

- Functions of composite variables:
  - "are more than half the inputs nonzero?" (majority)
    - RG yields a series of functions that are nonzero for a fraction of inputs $\propto 1/\sqrt{N}$
  - "is number of nonzero inputs divisible by 3?"
    - RG is nonzero for 2/3 of inputs (vs 1/2 for generic function)
  - Behavior upon renormalization reflects fact that functions depend on fixed combination of inputs

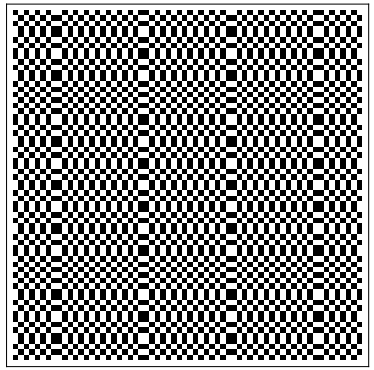
RG flow for low order polynomials is to constant, so they are not in generic phase

3rd order polynomial $f=a_0 \oplus \sum b_i x_i \oplus \sum c_{ij} x_i x_j \oplus \sum d_{ijk} x_i x_j x_k$



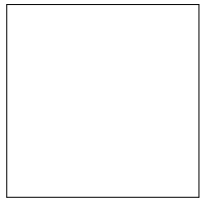
original function

N=14



after 2 renormalizations

N=12

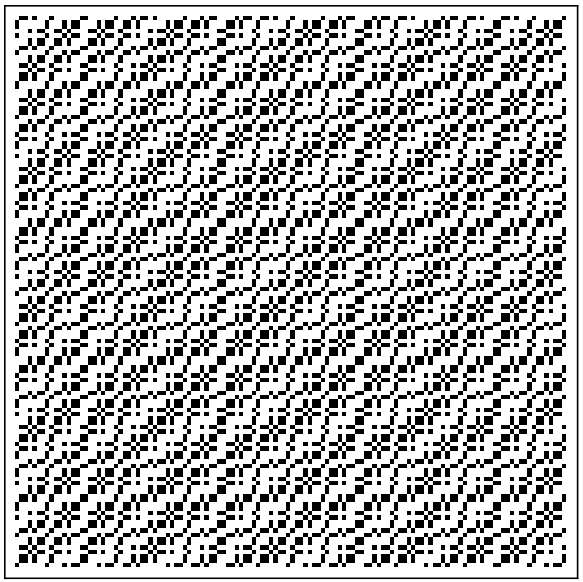


after 4 renormalizations

N=10

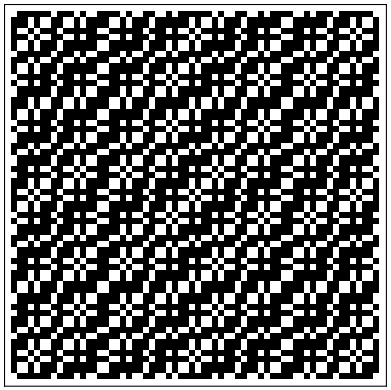
Some other functions do not “flow” to generic fixed point upon application of the RG

e.g., function whose value is one if the number of nonzero inputs is divisible by 3



N=14

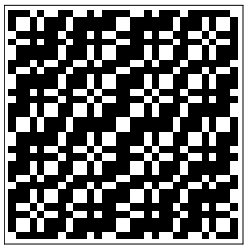
frac(black)=0.333



after 2 renormalizations

N=12

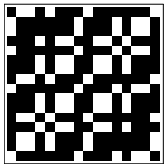
frac(black)=0.667



after 4 renormalizations

N=10

frac(black)=0.666



after 6 renormalizations

N=8

frac(black)=0.664

Possible relations between phases (defined by behavior upon repeated application of RG transformation) and computational complexity classes

Computational complexity: study of how computational resources needed to solve a problem grow with size of problem specification.

The complexity classes P and NP

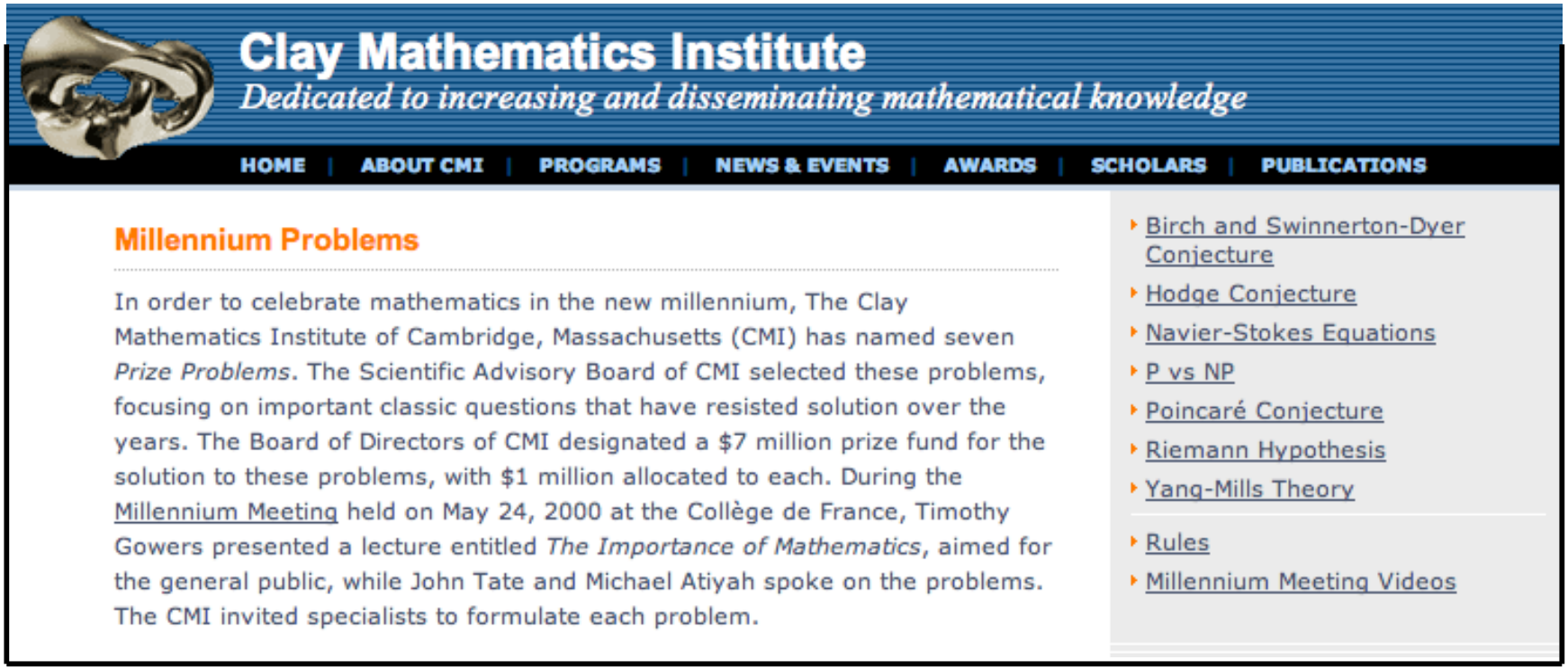
P: Problems that can be solved in a number of steps that grows no faster than polynomially with the size of the problem specification

NP: Problems for which a solution can be verified in a number of steps that grows no faster than polynomially with the size of the problem specification.

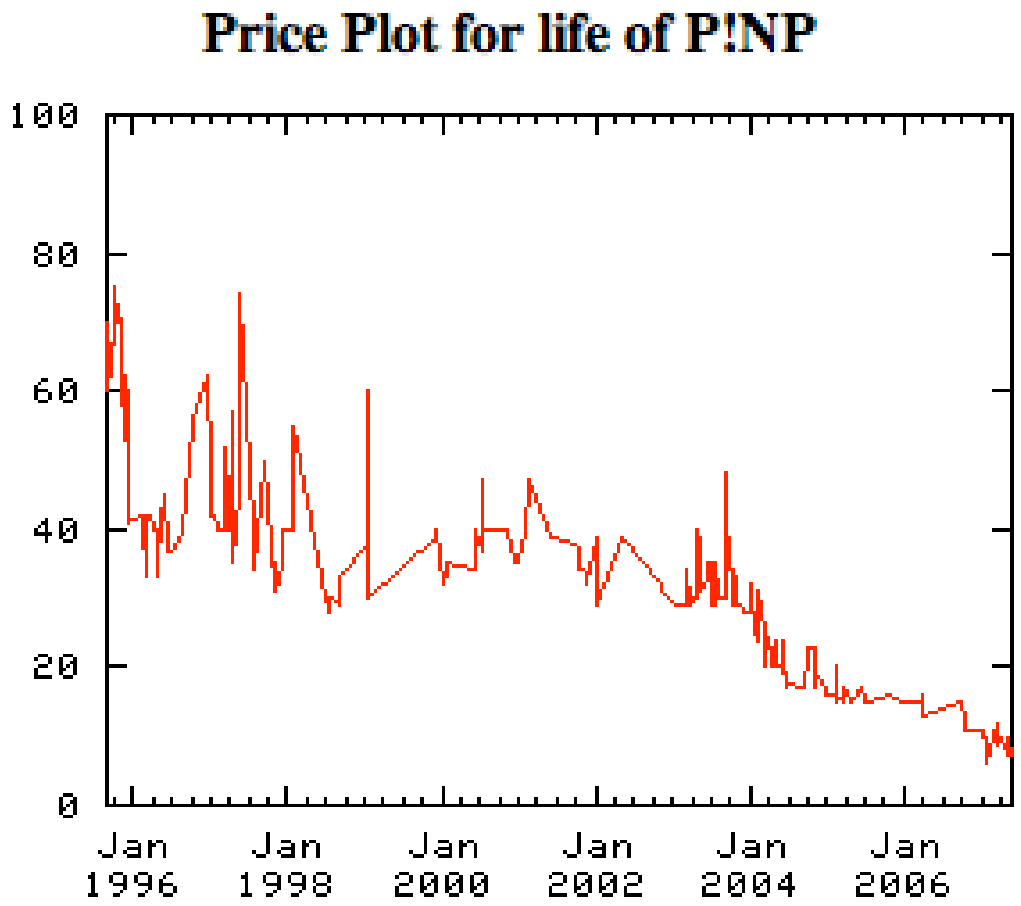
We know that problems in P are easy to solve.

We think that some problems in NP are hard to solve.

Whether or not P is distinct from NP is a key unanswered question in computational complexity



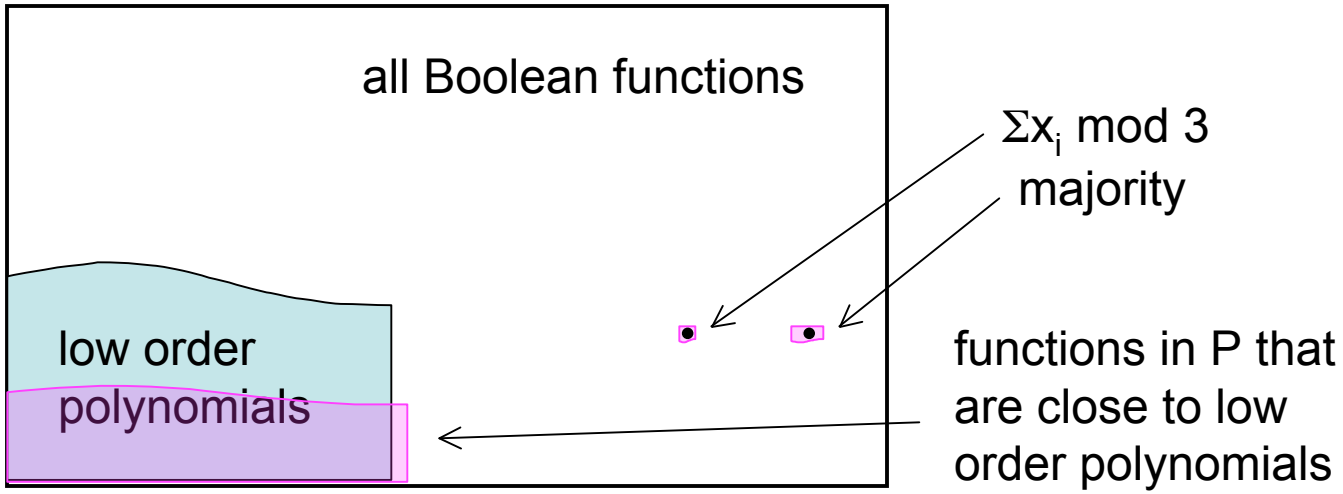
Market for P=NP or P≠NP proven by 2010



http://www.ideosphere.com/fx-bin/Claim?claim=P!NP

Possible relevance of renormalization group approach to characterizing P (problems that can be solved in polynomial time)

P is not a phase, but it is reasonable to conjecture that functions in P are either in or close to non-generic phases



Efficiently computable functions and phases do not coincide

- There are any more low-order polynomials (with order less than N^x with $x<1$) than there are efficiently computable functions

- A polynomial with a few terms of all orders up to N is efficiently computable, and appears to be in generic phase (but there is a non-generic function that yields the same output for all but a small fraction of input configurations)

Intuition underlying conjecture that efficiently computable functions are close to nongeneric phases

- Typical functions yield nonzero output on very close to half their input configurations
- Low order polynomials yield nonzero output on a significant fraction of their input configurations, but are nongeneric functions
- Products of O(N) variables yield nonzero output on an exponentially small fraction of input configurations
- Efficiently computable combinations of products and sums that result in functions that yield nonzero input on significant fraction of input configurations exist — conjecture is that restrictions on the combinations leads to resulting function being non-generic (true for examples such as MAJORITY, DIVISIBILITY MOD 3)

A big question

- Is it possible to prove that all functions in P are in or near non-generic phases?

Summary

- A renormalization group (RG) approach similar to the one used to characterize phase transitions in condensed matter systems can be used to classify Boolean functions.
  - RG identifies functions in a generic phase
  - RG identifies some functions that are in non-generic phases
  - P is not a phase, but it is a plausible conjecture that functions in P are either in or close to non-generic regions of the phase diagram.