Randomized Complexity

Warmup with some simple puzzles showing the slippery nature of probability.

- Two children: Each child is equally likely to be a boy or a girl. A family has 2 children. One of them is a boy. What is the chance that the other one is a boy?

(If the question said, the older one is a boy, then the answer would be 1/2.)

- Monty Hall: Game show with 3 doors. One has a car behind it, the other two have goats. You pick a random door, say # 1. Host then opens another door, say #3 and reveals a goat. You are then offered the possibility of switching to door #2. Should you?

One more puzzle

Two envelopes: you have the choice between two envelopes containing money. One envelope has twice as much as the other. You pick one. Then you're asked if you want to switch.
Should you?

Some examples where randomness seems useful…

- Cryptography: if an eavesdropper can predict what you're going to do, you've got a problem.
- Symmetry breaking: break up the "hallway dance" (useful in distributed computing).
- Monte Carlo simulation
- Testing polynomial equality
- Database checking:

The key idea we just saw

- Random fingerprinting: find a small random "fingerprint" of a large object. (e.g. value f(z) of a polynomial at a point z, in our first example).

- Example objects: strings, documents, data structures, etc.)

- The fingerprint captures essential information about the larger object: if 2 large objects are different, their fingerprints are usually different.

The big open question related to randomness

Do we need it?

Can we "derandomize" any randomized algorithm, I.e. convert it into a deterministic algorithm with roughly the same efficiency?

Randomized Complexity Classes

BPP -- Bounded Error Probabilistic Polynomial Time

Class of languages L for which there is a polynomial time algorithm M(x,r) such that for all inputs x:

- If x in L, then M(x,r) accepts with probability at least 2/3
- If x not in L, then M(x,r) accepts with probability at most 1/3

Using probabilistic TM -- has a tape containing random bits "r".

- The numbers 1/3 and 2/3 don't matter so much, because we can "amplify" the probability differences.

Randomized Complexity Classes

BPP -- Two-sided error

RP -- Randomized polynomial time

Class of languages L for which there is a polynomial time algorithm M(x,r) such that for all inputs x:

- If x in L, then M(x,r) accepts with probability at least 1/2
- If x not in L, then M(x,r) always rejects.

coRP (reverses the side of the error)

Some Relationships:

$P \subseteq RP$ $P \subseteq coRP$ $P \subseteq BPP$ $RP, coRP \subseteq BPP$
$RP \subseteq NP$ $coRP \subseteq coNP$ $BPP \subseteq PSPACE$

$BPP \subseteq NP$??? We don't know.
We can't even rule out BPP=NEXP! But many think BPP = P!