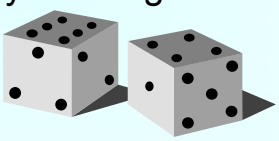
A Randomize Protocol for Consensus

A complete network graph (clique)
n - total number of processes.
f - total number of faulty processes.
Assumption: **n > 5f.**

This algorithm solves a more complex problem where the failure model is **Byzantine**, i.e. the failed processes can send arbitrary messages to arbitrary processes (may lie), or may fail.



The protocol (Ben-Or variation)

Round=0; x = initial value
Do Forever:
    Round = Round + 1
    Step 1
    Step 2

Step 1:
  Send Proposal(Round,x) to all processes
  wait for n-f messages of type Proposal(Round,*)
  if at least n-2f messages have the same value v
      then x = v (that value)
      else x = undefined

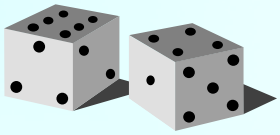
The Protocol (cont.)

Step 2:

Send Bid(Round,x) to all processes
wait for n-f messages of type Bid(Round,*)

v is the real value (0/1) occurring most often
and m is the number of occurrences of v

if m >= 3f+1
then **Decide** (x=v)
else if m >= f+1
then x = v
else x = **random** (0 or 1)



Submission: Thursday, October 27 at the beginning of class.
Solution must be typed. No collaboration is allowed.

Homework

1. Prove that the protocol in pages 20-22 is correct.
i.e. that it satisfies the agreement, validity and termination (with probability 1) requirements.
Termination means - for reaching a decision.
Assume at most f Byzantine failures.

The communication is reliable FIFO unicast, although messages from different processes can arrive in different order to different processes.

2. (only 437) Modify the algorithm so that eventually, all non faulty processes that decide also halt.
Try to make the modification as simple as possible.