Last we introduced Byz aggs.

Problem.

ETC alg: $f+1$ rds, $m > 3f$ proc, exp. comm. (bits)

Proof that $3f+1$ necessary.

Today go back + relate BA for binary values to arb. value set
Then consider connectivity issues + other problems related to consensus.

(And on to asych systems!)
Agreeing on a bit vs. agreeing on a more general value \( \text{(in V)} \)

Essentially the same problem!

It may seem that agreeing on a value from a big set, e.g. \( 10, \ldots, 99 \), might introduce its own complications.

But not really. Can use a \text{binary BA alg as a subroutine for general BA, with not much overhead: 2 extra rounds, \( \mathcal{O}( \log^2 n \log n \cdot n^2 ) \) bits}\) & few msgs (\( \mathcal{O}( \log^2 n \log n \cdot n^2 ) \) bits).

Says the core problem is simply agreeing—regardless of what we're agreeing on.

Again, need \( n > 3 \).

\textbf{Proc has } \{ x \} \text{ init value}

\text{1. intermediate value,}

\text{2. } \text{note - input for the binary "subroutine"}

\textbf{rd1: Send } x \text{ to all (incl self - can simulate locally, nice for uniformity)}

\text{Actually, need to do this in EIG alg, too}

\text{if in set received, there are } \geq m-f \text{ copies of some } v \in V,

\quad \text{then } y := v

\quad \text{else } y := \text{null}

\text{(only retain clear preferences)}

\textbf{rd 2: Send } y \text{ to all}

\text{if in set received, there are } \geq m-f \text{ copies of some } v \in V,

\quad \text{then note := 1}

\quad \text{else note := 0}

\text{Also, set } z := \text{the non-null value that occurs most often among received msgs. (ties arbitrary; if no such value, } z \text{ undefined)}.
Then run BA on the vote values.

If \( \text{decide } 1 \) is defined, \( \text{decision} := 2 \)
else \( \text{decision} := \text{default } v_0 \)

Key Lemma 1: At most 1 value \( v \in V \) is sent in rd 2, msg by nonfaulty proc.

\[ \begin{align*}
\text{Pf: Counting argument.} & \quad (\text{nonfaulty}) \\
\text{Suppose there are two, } & i, j \text{ send } v + w, \text{ resp.} \\
\text{Then } i \text{ receives } & \geq n - f \text{ rd 1 msg} \text{ cont } v \\
& j \text{ receives } \geq n - f \text{ rd 1 msg} \text{ cont } w \\
\text{But since } & i \text{ is faulty, } j \text{ receives } \geq n - 2f \geq f + 1 \text{ rd 1 msg} \text{ cont } v \\
\text{But } & (f + 1) + (n - f) > n, \text{ too many received msg.}
\end{align*} \]

\[ Q.E.D. \]

Theorem: Solves BA.

\[ \begin{align*}
\text{Pf: Termination:} & \quad \text{Obvious.} \\
\text{Validity:} & \quad \text{Easy. If all nonf. start with } v, \text{ all but } v \text{ at rd 1,} \\
& \text{all set } y := v \\
& \text{all set note } := 1, \ z := v \\
& \text{Then validity for subroutine implies decide } v.
\end{align*} \]

Agreement:

If sub. decides 0, then \( v_0 \) decided by all
If sub. decides 1, then

By validity for subroutine, some nonfaulty \( i \) must begin subroutine
with note = 1.
So processes receive \( \geq n - f \) rd 2 msgs cont. some particular \( v \).

So \( n - f \) rd 2 msgs cont. \( v \) from non-faulty processes.

So any other non-faulty \( j \) also receives \( \geq n - 2f \) msgs with \( v \) (from non-faulty processes).

Now by Lemma 1, only faulty processes can send value \( \neq v \) in rd 2.

So \( j \) receives \( \leq f \) msgs cont. values \( \neq v \), in rd 2.

\( n - 2f > f \), so \( v \) is the most frequent value.

So \( j \) sets \( z^i = v \).

This means \( j \) decides \( v \).

So, all nonf. decide \( v \).

This result can be thought of as showing a kind of "reducibility" between problems.

\[ \text{V-BA} \leq \text{Binary BA} \]

But I don't have a formal def for "reducibility".

Will see other "reducibilities" later in course—usually transformations of algorithms that run in one model so they can run under a weaker set of assumptions.
**General Graphs**

$m > 3f$ isn't the whole story.

Consider:

4, but can't tolerate 1 fault

Because center can disconnect.

The whole story is:

**Theorem:** BA reliable in $n$-node graph $G$ tolerate $f$ faults iff both:

1. $m > 3f$
2. $\text{conn}(G) > 2f$

defined to be min # of nodes whose removal results in either a disconnected graph or 1-node graph.

**Ex:**

- $\text{conn} = 3$
- $\text{conn} = 3$
- $\text{conn} = 1$

**Pf:** If $m > 3f$ and $\text{conn}(G) > 2f$

Then can simulate total connectivity alg.

The key is how to implement reliable comm. from a node $i$ to another node $j$.

Rely on Menger's Theorem, which says that a graph is $c$-connected (that is, has conn $\geq c$) iff each pair of nodes is connected by at least $c$ node-disjoint paths.  (see i & j in example 1 above)
So here \( \geq 2f+1 \) disjoint paths.

So send msg. send on all these paths.

Majority must be correct.

So take majority value.

Only if:

Already shown need \( n > 3f \).

Remains to show \( \text{conn}(G) > 2f \)

Just do simple case that gives the key idea:

\[
\text{conn} \leq 2 + f = 1 \quad \text{impossible}
\]

Canonical example:

![Diagram of 1, 2, 3, 4]

1 + 3 can be disconnected by removing 2 + 4 (2-connected).

2 copies:

![Diagram of 2 copies of 1, 2, 3, 4]

Consider 1, 2, 3. Looks to them like

So decide 0.

Consider 1', 2', 3': Decide 1

Consider 1', 4', 3': Must agree, contradiction.

As before, can generalize pf to conn \( \geq 2f \), or 2ccc reduciton.
Weak BA

It turns out that the bounds \( n > 3f + \text{const} > 2f \) are fundamental in consensus-style problems with Byz. failures.

E.g. in timed (partially sync) case, to do clock sync in presence of some number of Byz. faulty proc. same bounds hold.

Byz. fast keeping squad (HW)

Prop generally similar

E.g. consider weak agreement - base agreement, term as BA

But weaken validity to say:

If there are no faulty procs & all procs start with \( v \), then \( v \) is the only possible decision value.

A similar condition was used for the 1 (commit) case for the two generals problem.

Construct the situation where the decision has to go a certain way.

However, still impossible.

Though now harder to prove impossibility

pf: If follows from ordinary BA

Only if: first do 3 no 1 as ex 2 if no f then \( \exists \) LTR connectivity

Assume FA, proc 1, 2, 3

\( \exists \) execs in which all start with \( \{0\} \) no failures

Both result in decisions.
Let \( p = n/2 \) on rounds for all votes to decide, in both.

5: Paste 2n copies of \( A \) into \( n \)-process ring

\[
\begin{array}{cccc}
1 & 2 & 3 & 1 \\
2 & 3 & 1 & 2 \\
3 & & & \text{exec} \\
\end{array}
\]

Start top half with 0
Bottom half with 11

Can show any 2 adjacent pairs in \( S \) must agree, in \( S \) exec
(Because they must agree in \( A \))

So all agree.

WLOG say decide 1.

Now consider block of \( 2n+1 \) consecutive 0s.
Claim all but ends indist. from all all-0 exec \( \text{exec} \) for 1st rd

\[
\begin{array}{cccc}
1 & 1 & 2 & \text{at ends """""} \\
11 & 2 & \text{at ends """""} & 20 & 20 \text{ rd} \\
\end{array}
\]

Mid indistinguishable for \( \text{for} \) \( \text{rd} \) for \( \text{rd} \).

But that's enough for it to decide 0, contradicts the fact that it decided 1.
Ma on consensus in sync model

Some variations on this problem that have been studied

\{ k-agreement \} (can have k values, not just 1)
\{ approx. agreement, commit \}

I'll just cover quickly in class.

k-agreement

Generalizes ordinary stopping agreement by allowing \( k > 1 \) different decisions instead of just one

Practical motivation?

For resource allocation and network problems.

E.g. agree on small \( k \) of least frequencies for receiving heats.

Really motivated by math considerations:

Elegant theory + interesting things sounds.

Prob. statement

\( m \)-mode complete undirected, stopping failures
input\( \), decisions in \( V \)

agreement: \( \exists \) subset \( W \subseteq V \), \( |W| = k \), all dec. values in \( W \)

validity: Any decision value is initial value of some process.

(That corresponds to stronger validity I mentioned for)

1-agreement.

termination: All nonf. eventually decide

Alg. Blood-Min (like Blood-Set for ordinary)

agreement
Each maintains min value seen, initially own value.
Decide on final min value.
If do this for $f+1$ rounds, solves ordinary stopping agreement
(1-agreement)
For k-agreement, just do for $\left\lfloor \frac{f}{k} \right\rfloor + 1$ rounds.

Not obvious - allowing k values divides the runtime
by k.1

Correctness:
\[ M(n) = \text{set of min vals of active proc after } n \text{ rounds} \]
This set can only decrease over time.
(subset sense)

Lemma 1: $M(n) \subseteq M(n-1)$ for all $n$, $1 \leq n \leq \left\lfloor \frac{f}{k} \right\rfloor + 1$.
Proof: Any min value after $n$ is someone's after $n-1$

Lemma 2: If at most $d-1$ processes fail during rd $n$, then
$|M(n)| \leq d$.
(at most d different values left, if $d-1$ fail)
Generalizes result for $d=1$: if more fail then all same.
Proof: Suppose for contradiction that $\sum m \leq d-1$ failures
$|M(n)| > d$.

Let $\sum m = \max(M(n))$
\[ m' = \text{any other, } m' < m \]
\[ m' \text{ is an element of } M(n) \]
Then \( m' \in M(n-1) \) by Lemma 1; let \( i \) be a process active after 
\( n-1 \) with \( m' \) as its min.

If \( i \) does not fail in rot \( r \) then every process receives \( m' \).

Contrad. the fact that some active process has \( m > m' \) as its min.

So, \( i \) must fail.

But this is true for every value of \( M(r) \neq m \): There

So there must

must be some process active after rot \( n-1 \) with that initial value, who fails in rot \( r \).

So, there must be > \( k \) failures.

Contrad. 

\( \square \)

3.2m: Correct

Pf: Lem., validity easy

Agreement: By contradiction. Suppose in an exec with \( k \) failures, there > \( k \) decision values.

Then # of min-val’s for active gives after \( \frac{n}{k} + 1 \) rounds

is \( \geq k + 1 \).

That is, \( |M(\lfloor \frac{n}{k} \rfloor + 1)| \geq k + 1 \).

Then by Lemma 1; \( |M(r)| \geq k + 1 \) for all \( r \), \( 0 \leq r \leq \lfloor \frac{n}{k} \rfloor + 1 \)

(see only shrink)
Then by Lemma 2, at least k processes fail in each round.

Total of \( \left\lceil \frac{f}{k} \right\rceil + 1 \) failures

\( \leq \) failures in each rd

\# of rd's

\( > f \), control

---

**Lower bd**

Quick sketch.

Interesting result statement: \( \left\lceil \frac{f}{k} \right\rceil + 1 \) is exact lower bd.

Interesting techniques: Though there is general theory for these & further applications of them are beyond the scope of this class.

Assume \( |V| \geq k+1 \) (else problem is trivial).

Recall pf of \( f+1 \) - round lower bd for 1- agreement.

Chain of execs, for assumed alg.

For same allowed decisions, a simple chain doesn't work: Can't say unique decision value in each exec., same in 2 consecutive:

\[ a_i \stackrel{p}{\rightarrow} a_{i+1} \]

\( dec = v \quad dec = v \)

E.g., for \( k = 2 \), could have 2 different values here without violating agreement.

However, a more complex, \( k \)-dimensional structure will work.
Proof impossibility:

Assume: \( n \)-process alg. for \( k \)-agreement, \( k \leq n \) failures.

Halt in \( n \leq \left\lceil \frac{1}{k} \right\rceil \) rounds.

\( n \geq f + k + 1 \) (at least \( k+1 \) procs don't fail in any exec).

All nonf. procs decide at end of rd \( n \).

All send to all at all rounds.

\( V = \{0, \ldots, k\} \)

Show contr. by producing exec. with \( \geq k+1 \) different decisions.

If sketch: use \( k \)-dim collection of execs rather than 1-dim. adjacent execs induct. to some nonf. proc.

Bermuda \( \Delta \): (algorithm vanishes)

\( k = 2 \)

Nodes correspond to execs.

In general, for arbitrary \( k \), use "\( k \)-dim simplex" (alg. topology).

1-dim simplex = line

2-dim = \( \Delta \)

3-dim = tetrahedron, etc.

\( \Delta \) grid points, connect to give little \( \Delta \)s (triangulation).

Extrudes to \( k \) dims.

How assign exec to each vertex:
Conner: No failures, all start with same value.

Edges (+ higher - chain "faces"): Suit values chosen from those of
the corners of the face. Can have failures.

Interior: Mixture of all inputs.

Assignment done in such a way that: The execs morph gradually
as we move in each direction:

just [remove or add a new as before.
fail a recover a process]

change initial value

Also, to each vertex:

assign index of some process nonf. in execution of that of vertex.

Do this "consistently."

So that for every tiny $\Delta t$ there exists execution $\alpha$ with $\leq f$ faults
that is "compatible" with execs + procs assigned to its corners;

1. all these procs are nonfaulty in $\alpha$

2. if $(z, i)$ assigned to some corner, then $\alpha \sim z'$

Ex: Suppose do this for our previous chain arg:

\[ \cdots \rightarrow \pi_c \rightarrow \pi_{c+1} \rightarrow \cdots \]

Let $\pi_i$ nonf. in $z_i + z_i^{+1} \rightarrow \pi_{i+1}$ nonf. in $z_{i+1} + z_{i+2}$, etc.

Assign $\hat{\pi}_i$ to $z_i$, $\hat{\pi}_{i+1}$ to $z_{i+1}$, etc.

Let the "new exec" assigned to

\[ z_i + z_{i+1} \rightarrow \beta = z_{i+1} \]

Then $\pi_i \beta \hat{\pi}_{i+1}$ both nonfaulty in $z_{i+1}$.

Also, $\beta \preceq \hat{\pi}_i$ and $\beta \preceq \pi_{i+1}$.
This assignment requires the assumption \( r \leq \left\lfloor \frac{n}{k} \right\rfloor \), that is, 
\( f \geq r k \).

Why: Construction requires us to fail \( r \) processes (1 per sol) to construct the chain in each dimension \( k \) dimensions, so \( r k \) failures.

More complicated, but basically same idea as chain count.

Now color each vertex with a "color" in \( \{0, \ldots, k\} \),
\[ \text{color}(x, c) = i \] decision value in \( i \)

Properties:

1. Colors of \( k+1 \) corners of any \( k \)-simplex \( \Delta \) all distinct
   (by validity)

2. Color of each pt on any edge (or higher-dim face) is the color of one of its corners.
   (by validity)

Now pull a math trick out of a hat:

Coloring of a triangulated simplex having these 2 properties has a special name in algebraic topology: Spencer Coloring.

Spencer's Lemma says any such coloring has some tiny simplex whose \( k+1 \) corners are colored by all \( k+1 \) colors.

But here this yields an execution with \( k+1 \) values.

Contradicts \( k \)-agreement!