Contract signing

- Two parties want to agree on a contract
  - each will sign if the other will sign, but do not trust each other
  - there may be a trusted third party (judge)

    but it should only be used if something goes wrong
- In real life: contract signing with pen and paper
  - sit down and write signatures simultaneously
- On the Internet...
  - how to exchange commitments on an asynchronous network?
  - “partial secret exchange protocol” due to

    Even, Goldreich and Lempel [EGL85]

Contract signing – EGL protocol

- Partial secret exchange protocol for 2 parties (**A** and **B**)
- **A** (**B**) holds 2N secrets $\mathbf{a_1},\ldots,\mathbf{a_{2N}}$ ($\mathbf{b_1},\ldots,\mathbf{b_{2N}}$)
  - a secret is a binary string of length L
  - secrets partitioned into pairs: e.g. $\{(\mathbf{a_i}, \mathbf{a_{N+i}}) \mid i=1,\ldots,N\}$
  - **A** (**B**) committed if **B** (**A**) knows one of **A**'s (**B**'s) pairs
- Uses "1-out-of-2 oblivious transfer protocol" **OT(S,R,x,y)**
  - **S** sends **x** and **y** to **R**
  - **R** receives **x** with probability ½ otherwise receives **y**
  - **S** does not know which one **R** receives
  - if **S** cheats then **R** can detect this with probability ½

Contract signing – EGL protocol

(step 1)

for (i=1,…,N)

OT(A,B,a_i,a_{N+i})

OT(B,A,b_i,b_{N+i})

(step 2)

for (i=1,…,L) (where **L** is the bit length of the secrets)

for (j=1,…,2N)

A transmits bit **i** of secret a_j to **B**

for (j=1,…,2N)

B transmits bit **i** of secret b_j to **A**

Contract signing - Results

- Modelled in PRISM as a DTMC (no concurrency) [NS06]
- Discovered a weakness in the protocol:
  - party **B** can act maliciously by quitting the protocol early
  - this behaviour not considered in the original analysis
- More details:
  - if **B** stops participating in the protocol as soon as he/she has obtained at least one of **A** pairs, then, with probability 1, at this point:
    - **B** possesses a pair of **A**'s secrets
    - **A** does not have complete knowledge of any pair of **B**'s secrets
  - Protocol is therefore not fair under this attack:
    - **B** has a distinct advantage over **A**

Contract signing - Results

- The protocol is unfair because in step 2: **A** sends a bit for each of its secret before **B** does.
- Can we make this protocol fair by changing the message sequence scheme?
- Since the protocol is asynchronous the best we can hope for is with probability ½ **B** (or **A**) gains this advantage
- We consider 3 possible alternate message sequence schemes...

Contract signing: EGL2

(step 1)

...

(step 2)

for (i=1,...,L)

for (j=1,...,N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (j=1,...,N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=N+1,...,2N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (j=N+1,...,2N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

Contract signing: EGL3

(step 1)

...

(step 2)

for (i=1,...,L) for (j=1,...,N)

A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (i=1,...,L) for (j=N+1,...,2N)

A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

Contract signing: EGL4

(step 1)

...

(step 2)

for (i=1,...,L)

A transmits bit **i** of secret $\mathbf{a_1}$ to **B**

for (j=1,...,N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=2,...,N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

for (i=1,...,L)

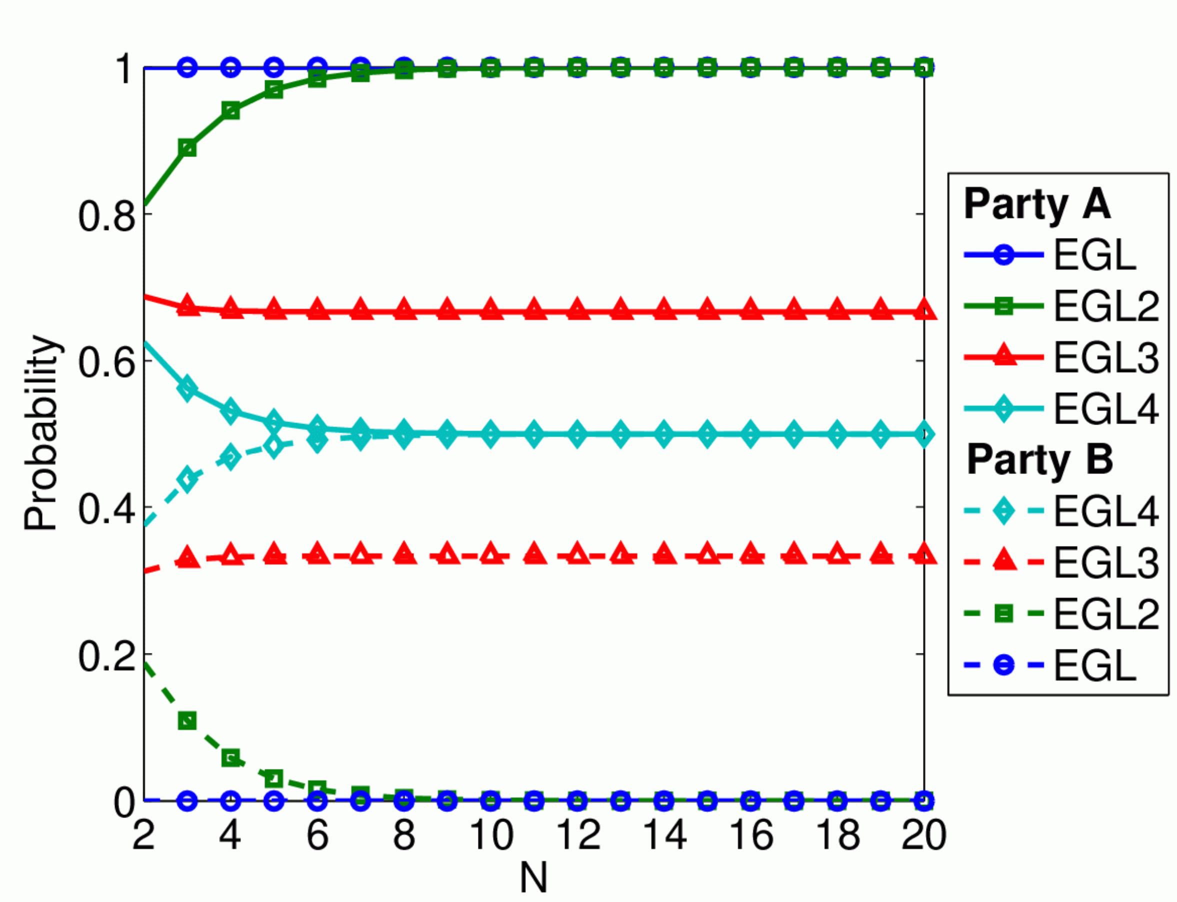
A transmits bit **i** of secret $\mathbf{a_{N+1}}$ to **B**

for (j=N+1,...,2N) B transmits bit **i** of secret $\mathbf{b_j}$ to **A**

for (j=N+2,...,2N) A transmits bit **i** of secret $\mathbf{a_j}$ to **B**

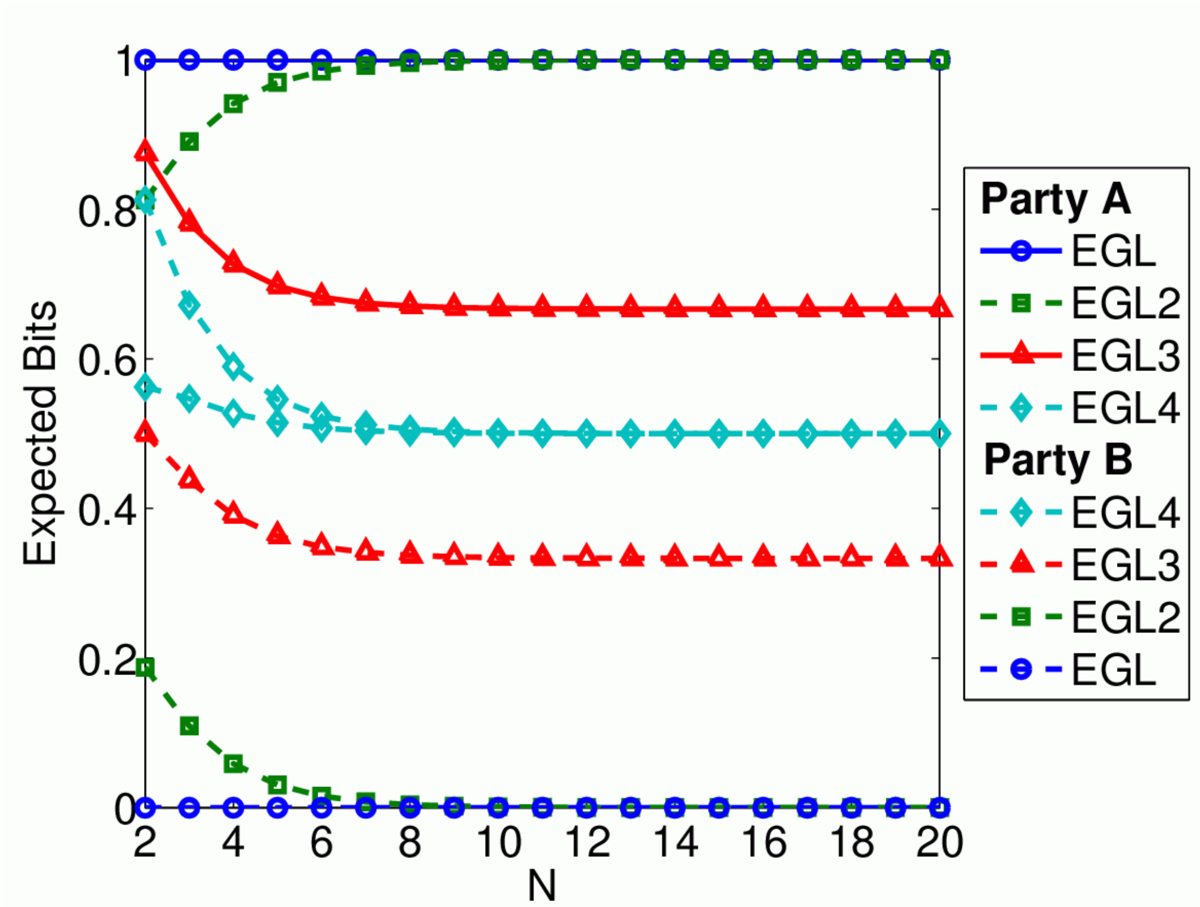
Contract signing - Results

- Probability that the other party gains knowledge first
  (the chance that the protocol is unfair)



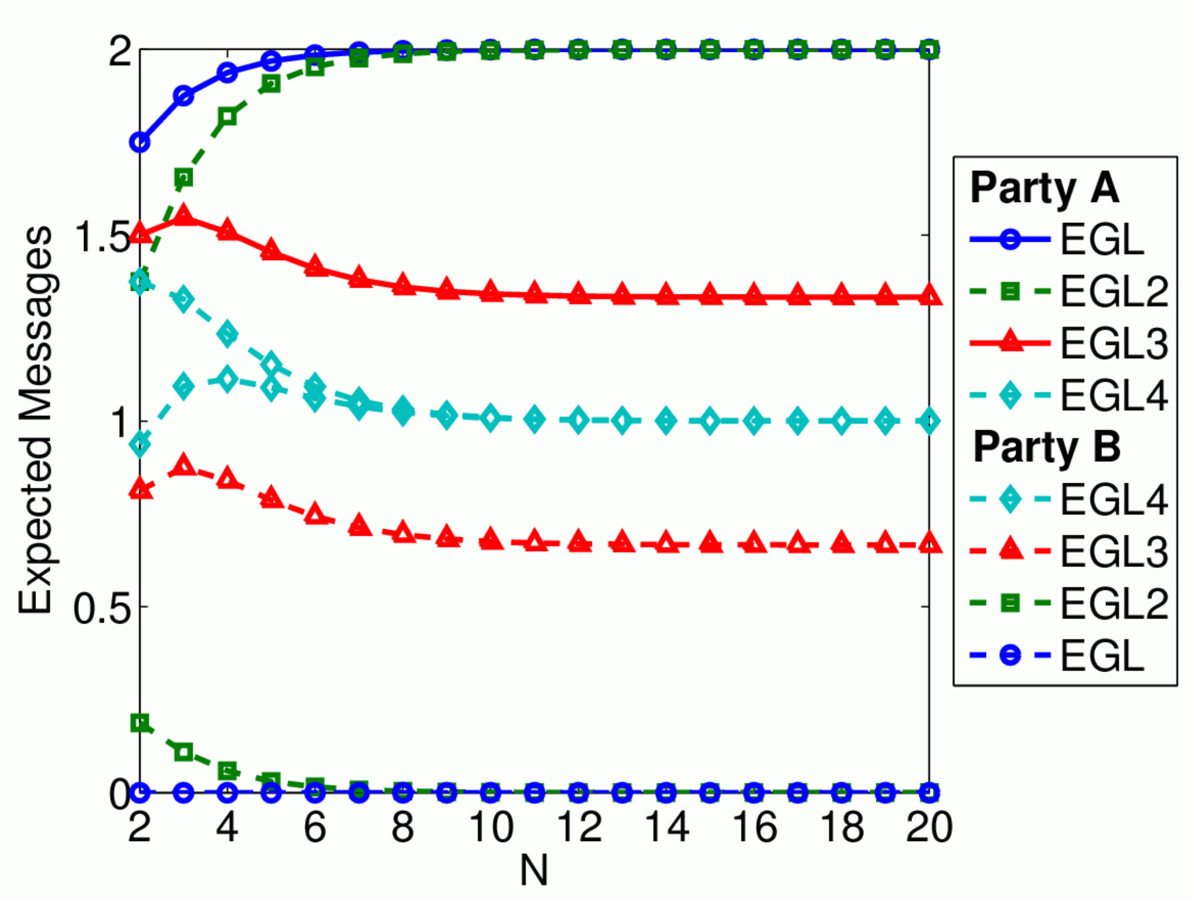
Contract signing - Results

- Expected bits a party requires to know a pair once the other knows a pair (quantifies how unfair the protocol is)



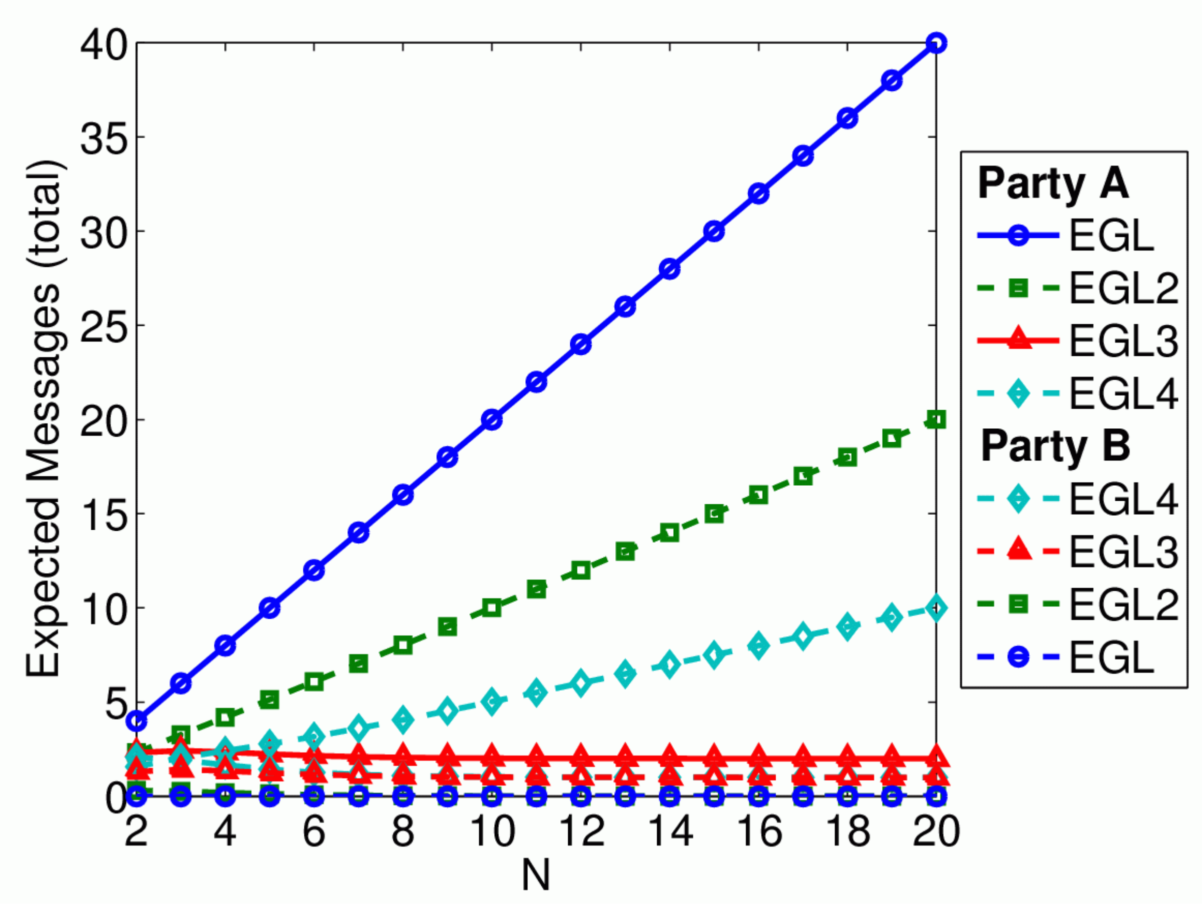
Contract signing - Results

- Expected messages a party must receive to know a pair once the other knows a pair (measures the influence the other party has on the fairness, since it can try and delay these messages)



Contract signing - Results

- Expected messages that need to be sent for a party to know a pair once the other party knows a pair (measures the duration of unfairness)



Contract signing - Results

- Results show EGL4 is the ‘fairest’ protocol
- Except for duration of fairness measure:

  Expected messages that need to be sent for a party to know a pair once the other party knows a pair

  - this value is larger for **B** than for **A**
  - in fact, as **N** increases, it increases for **B**, decreases for **A**
- Solution: if a party sends a sequence of bits in a row (without the other party sending messages in between), require that the party send these bits as as a single message

Contract signing - Results

- Expected messages that need to be sent for a party to know a pair once the other party knows a pair (measures the duration of unfairness)