Effective Competition Organization to Obtain Effective Algorithmic Solutions from a Motivated Community

Karl Lieberherr, Northeastern University,
College of Computer and Information Science, PRL
joint work with
Ahmed Abdelmeged (PhD Dissertation)
Ruiyang Xu

Solving Complex Problems using the Crowd

We want **Incentive Compatibility**.

Some are better than others: we want a **fair, peer-based meritocracy**.

Some want to game the system by **colluding/lying.**

Complex behavior of the crowd!

- The correct solution is not known.
- Correct solutions are refuted.
- Incorrect solutions are defended.
- Players may change their mind about which solution is correct.
- Players may lie about their strength and intentionally help a friend to become more meritorious (collusion).
- **How can we get a useful signal out of this complex behavior?**
  - **a data mining problem.**
  - **wrong approach: accumulate information about solutions.**
  - **better approach: accumulate information about players.**

Contributions

- We offer
  - A new kind of game: called a side-choosing game for modeling formal science communities (Algorithms is a good Example).
  - A theoretical study of side-choosing games
    - 2 Theorems:
      - A Representation Theorem that provides a family of desirable ranking algorithms satisfying three axioms, including the collusion-resistance axiom.
      - Incentive-Compatibility Theorem: shows that collusion-resistance is essential for incentive compatibility.
  - A practical application of side-choosing games
    - Reports about how we use side-choosing games for course home works to take advantage of peer teaching.

Effective Organization of a Motivated Community to Obtain Effective Algorithmic Solutions

- In economics, when a person must rely on others to solve a problem there are two important constraints
  - Participation Constraint
    - ensures that people want to participate.
  - Incentive Compatibility Constraint
    - ensures that people are motivated to behave in a manner consistent with the best solution.

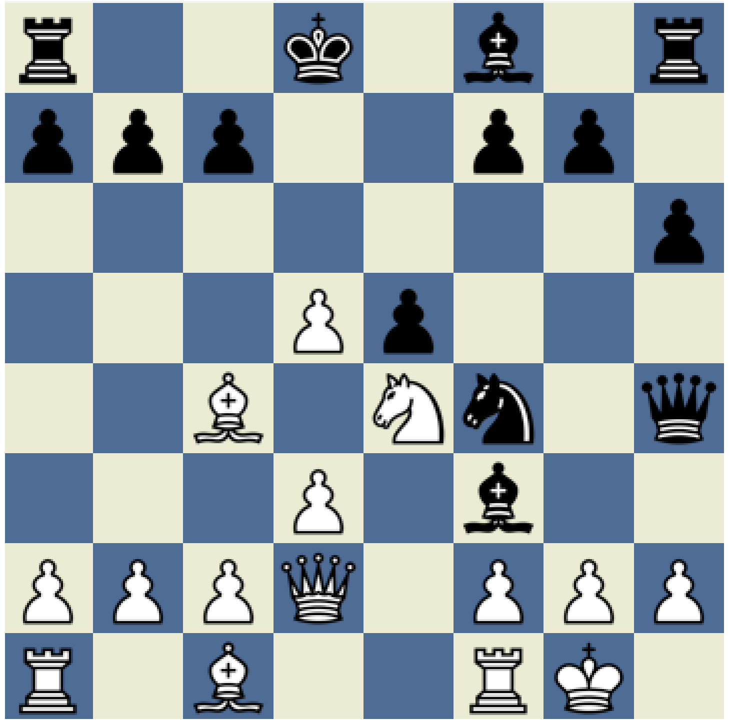
Our solution for both constraints: Side-Choosing Games with the right Ranking Mechanism

Motivating Example for Side-Choosing Game

- Choose a combinatorial game: Chess.
- Take a chess position P.
- Claim:
  - There exists a k so that Black, moving first from P, will check-mate White after k moves and
  - k is minimum.
- Abbreviated claim: Black mates White in at most k moves.

Organization is based on side-choosing games. What is a side-choosing game (SCG)?

Black mates White in at most k moves.



A claim about a combinatorial game G, e.g., Chess.

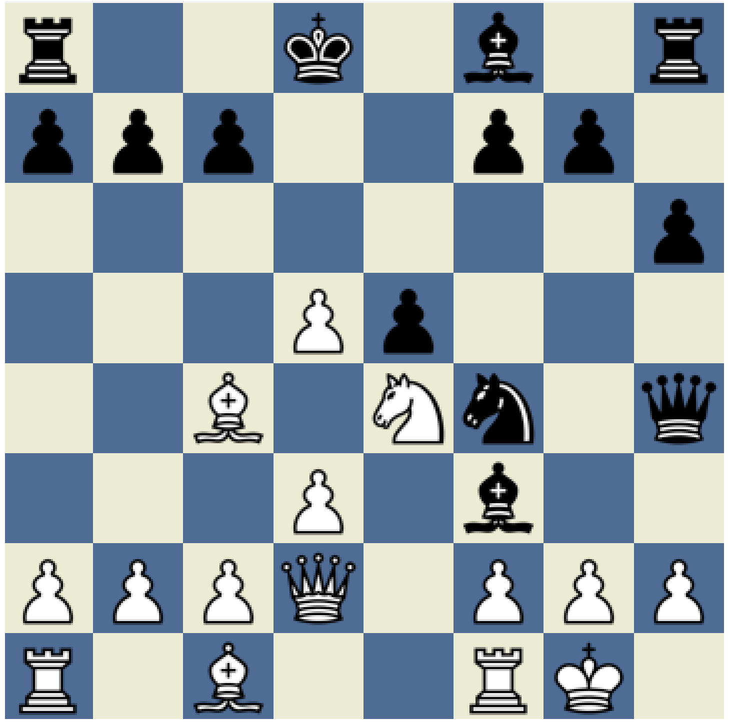
There is always a winner and a loser. No ties.

We ask 2 players x and !x:
x is a Proponent
!x is an Opponent

x and !x must defend their side-choice by winning the game.

What is a side-choosing game?

Black mates White in at most k moves.



We ask 2 players x and !x:
x is a **Proponent**
!x is a **Proponent**

We have one of them play as devil's advocate. Say x is devil's advocate. If x wins, !x was not a serious Proponent. Devil's Advocate = **Forced**.

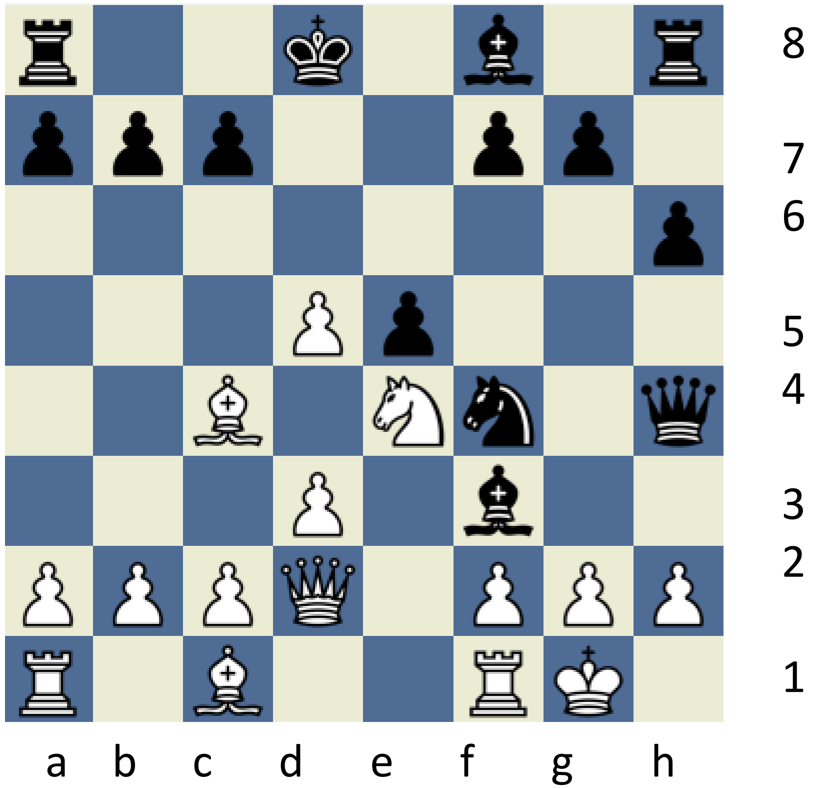
To play this SCG

Claim:

There exists a k so that Black, moving first, will check-mate White after k moves and k is minimum.

W	L	F
?	?	?

W: Winner
L: Loser
F: Forced



Proponents?
Opponents?

Another Issue: Distributing the Evaluation Work.

- Administrator: Defines claim/game; checks that rules are followed; determines who wins and loses and keeps track of results.
- Does Administrator have to solve the problems = develop winning strategies for claims?
- No! We want the administrator to be a **referee** who is interested in problem solutions but who wants to get them from the players.
- How can we make sure that the **peer evaluation** is **fair**?

Another Motivating Example: Software Development: specifying a function

- Pre and Post conditions for requirements
- Exists gcd in Function(Nat,Nat -> Nat) ForAll x,y in Nat Exists d in Nat:
  - d=gcd(x,y) $\wedge$
  - divides(d,x) $\wedge$ divides(d,y) $\wedge$
  - ! Exists s in Nat ((s>d) $\wedge$ divides(s,x) and divides(s,y)) $\wedge$
  - if x,y < C then Runtime(gcd,(x,y)) < RC
- E.g., C = 10^{10}, RC = 10 milliseconds.
- We are hoping for Euclid's or Schoenhage's algorithm.

Gamification of Software Development for Computational Problems

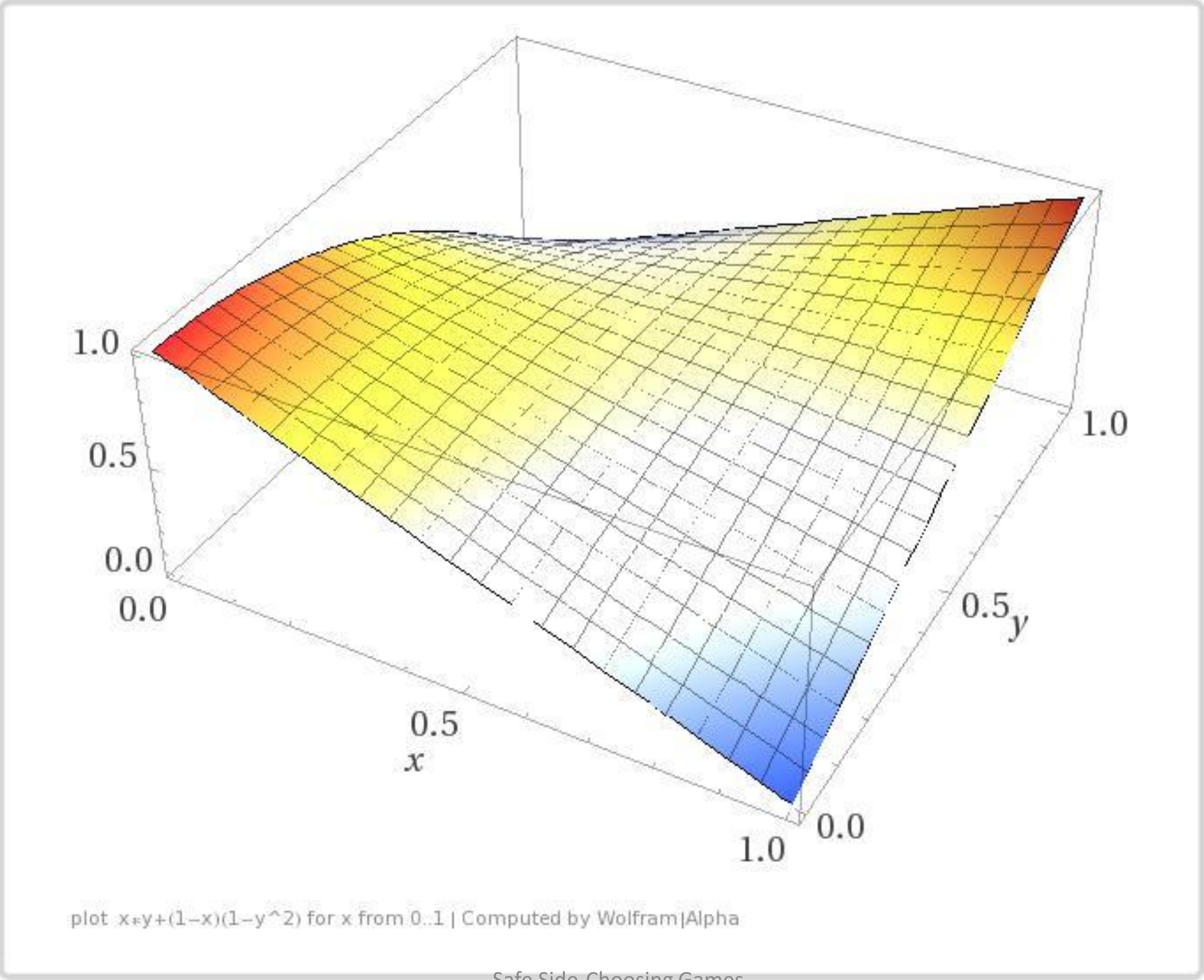
- Want reliable software to solve a computational problem? Design an SCG lab where the winning team will create the software you want.
- A lab consists of Game/logical sentence interpreted in some structure.

Formal Science Claims: Saddle Point / Silver Ratio

claim

G(c) = ForAll x in [0,1] Exists y in [0,1]: x*y + (1-x)*(1-y^2) >= c

- Strategy chosen depends on c.
  - G(0.5)
  - G(0.615)
  - G(0.616)


1.0
0.5
0.0
0.0
0.5
x
1.0
0.0
0.5
y
1.0
plot x*y+(1-x)(1-y^2) for x from 0..1 | Computed by Wolfram|Alpha

What is common to

- Chess Puzzle claim
- Software Development claim
- Formal Science claim
  - Two views
    - Combinatorial Game G
      - Claim: Start position is winning.
    - Logical formula ϕ interpreted in structure $\boldsymbol{A}$

Logical Formula Claim$\langle \phi, \boldsymbol{A} \rangle$

- Claim$\langle \phi, \boldsymbol{A} \rangle$
    - ϕ is a well-formed formula.
    - $\boldsymbol{A}$ is a structure, often consisting of several substructures. Think of $\boldsymbol{A}$ as a collection of data types that are needed to define the claim.
        - ϕ refers to the functions defined in those data types.
- What goes into ϕ, what goes into A? Into ϕ: what is better done by humans; into A: what is better done by machines.

Outline for the Talk

- Theory: Develop ranking theory for side-choosing games to find the most meritorious player.
  - Novelties:
    - Side-Choosing games.
    - Meritocracy Management for Side-Choosing Games: Ranking functions for side-choosing games. Map game results to a ranking of players.
  - Axiomatic approach:
    - Formulate desirable axioms for ranking functions.
    - Find representation theorem for ranking functions satisfying axioms.
- Applications/Results:
  - Lower barrier of entry for competition designers and participants
    - Simplify work of administrator
    - Make participation fun (collaborative, learning component)
  - Organize communities for experience-based learning
  - Software Development

Yaron Singer's Course: CS 284r: Incentives and Information in Networks

- Premise: Human Interactions follow mathematical patterns that can be leveraged by algorithms for acquiring, disseminating, and learning information.

Connection to Yaron’s Premise

- We study engineered networks where participants need to follow a protocol of interaction about claims.
- We study algorithms that take the results of those constrained interactions as input and produce
  - information about the participants and, indirectly through the participants,
  - information about the claims.
- We study techniques to engineer networks that disseminate specific knowledge through a network.

Connection to Yaron's Work

- So while Yaron gathers information from networks "in the wild" we gather information from networks we have constrained by rules. Our rules are motivated by observing how formal scientific communities work.

Definition: Side-Choosing Game

- A side-choosing game is a triple (G,SC,AA), where G is a two-person, draw-free, combinatorial game, SC is a side-choice configuration and AA is an agreement algorithm.
- G, SC and AA are defined separately. The important component is the combinatorial game G; SC and AA offer variation possibilities to define the side-choosing games. We will use simple instances of SC and AA for our side-choosing games.

Combinatorial Game Definition

1. There are two players. Sequential (turn-based). Perfect-Information, no chance or hidden information. (But players may hide their winning strategies.)
2. There is a finite set of possible positions of the game.
3. The rules of the game specify, for both players and each position, which moves to other positions are legal moves.
4. The game ends when a position is reached from which no moves are possible. A predicate on the final position determines who has won. There is an absolute winner: the first player to fulfill the winning condition. No ties or draws.
5. The game ends in a finite number of moves.

Terminology

- Design-time decision for Proponent/Opponent. This is the design time for the winning strategy.
- Run-time decision for Proponent/Opponent. This is the decision used when the defense game executes and might involve forcing at most one participant.
- Side-choosing game consists of two parts
  - simultaneous design-time decision
  - run-time decision (imposed from outside) followed by sequential games defending run-time decision.

Side-Choice Configuration Definition

- The side choices are made by the players but the Side-choice Configuration (SC) defines the sequencing of the design-time decisions.
- We have two players, x and y, who make a choice dx and dy for the start position of the combinatorial game G. dx, dy are elements of {P,O}. If x chooses dx=P then x claims to win the combinatorial game G from its start position. If x chooses dx=O then x claims to prevent y from winning G from its start position. The players make the side choices but SC specifies the configuration. Examples for SC:
    - simultaneous
    - sequential
        - with probability p for x to be the first player.
        - other context-sensitive mechanisms to choose the first player.

Agreement Algorithm Definition

- The agreement algorithm maps two players x,y with their design-time side choices dx=dy for combinatorial game G into a set of plays between the two players with, for each play, run-time side choices rx, ry such that at most one player is forced.
  - Example: Agreement algorithm CAA (Competitive Agreement Algorithm):
    - Randomly choose z, one of the players x,y and force z.
      - Play SCG(G, x, y, dz, d!z, !dz, d!z); z is forced
      - Play SCG(G, x, y, dz, d!z, dz, !d!z); !z is forced
    - Play SCG(G, x, y, dx, dy, rx, ry) is a game between x and y, where x makes design-time choice dx and y makes design-time choice dy. The run-time choices are rx and ry. If x is forced then rx=!dx and ry=dy. If y is forced then rx=dx and ry=!dy. The claim is: in the start position of G the player in the P role (d*=P) has a winning strategy.
    - dx,dy,rx,ry have values in {P,O}. z has values in {x,y}. !z =x if z=y. !z = y if z=x. !P=O, !O=P.

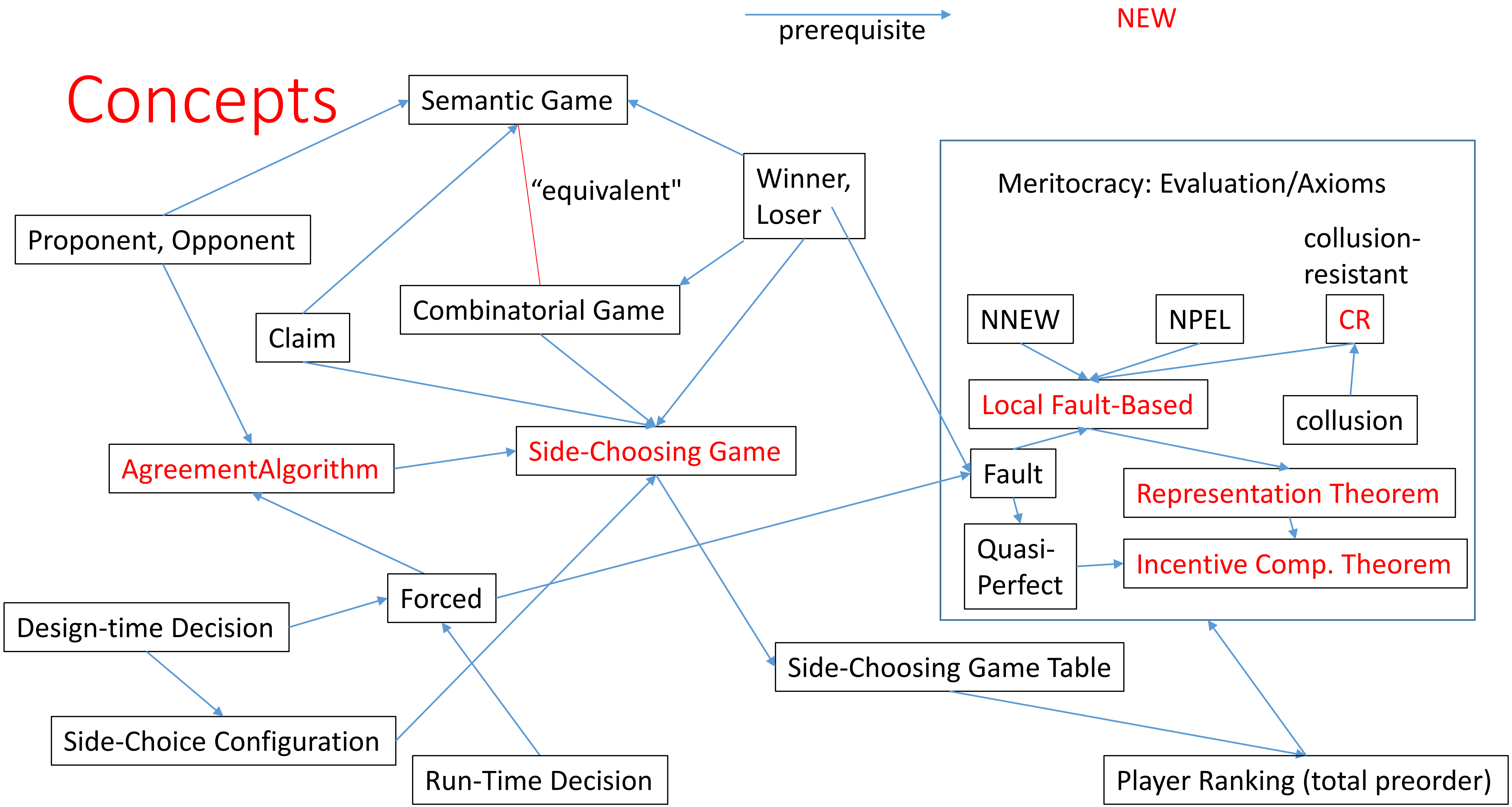
Motivation for CAA

- For some claims it is a disadvantage to have to move first because you give away a “secret” (e.g., the chess example). Therefore, we choose the forced player randomly to balance the potential disadvantage.
- We play two games to give each player a chance to test the other.

Examples of Combinatorial Games

- Truth of sentences in various logics (semantic games: Hintikka)
  - First-order predicate logic
  - Second-order predicate logic
  - etc.
- Positions in 2-person extensive-form games with perfect information. Choose a node (position) and ask: is it winning? Example: Chess.
- From Combinatorial Game Theory: Go, Checkers, Nim (solved), etc.

Concepts



prerequisite
NEW
Semantic Game
"equivalent"
Winner, Loser
Proponent, Opponent
Claim
Combinatorial Game
Side-Choosing Game
AgreementAlgorithm
Forced
Design-time Decision
Side-Choice Configuration
Run-Time Decision
Side-Choosing Game Table
Meritocracy: Evaluation/Axioms
collusion-resistant
NNEW
NPEL
CR
Local Fault-Based
collusion
Fault
Representation Theorem
Quasi-Perfect
Incentive Comp. Theorem
Player Ranking (total preorder)

benefits: Objective, Low Overhead, Correct, Targeted Feedback, Well-Defined Interfaces.

Benefits of Side-Choosing Games

- Objective: The result depends on how well the participants solve the computational problems coming from the claim/game.
- Low Overhead on Administrator: Prepare claim/game and check that the game rules are followed.
- Correct: The winners demonstrate their opponent’s lack of skills for current claim/game.
- Targeted Feedback: game plays give losers specific feedback.
- Participants interact through well-defined interfaces. Choosing side and following game protocol.

Toy Example: SCG Trace

✓ Proponent
✗ Opponent

SCG(∀x ∈ [0,1]: ∃ y ∈ [0,1]: x + y > 1.5, ✓, ✗)

✗ Provides 1 for x | Weakening (too much!)

SCG(∃ y ∈ [0,1]: 1 + y > 1.5, ✓, ✗)

✓ Provides 1 for y | Strengthening

SCG(1 + 1 > 1.5, ✓, ✗) ✓ Wins

SCG = Side-Choosing Game
SCG = Scientific Community Game
SCG = Specker Challenge Game
SCG = SemantiC Game
all are meaningful

Abstraction Barriers/Important Steps

- Widen: Semantic Games – Combinatorial Games: Generalize
  - Key to side-choosing game definition
  - Simplifies Meritocracy Theory for Side-Choosing Games
- Minimum Recording: Side-Choosing Game Plays – SCG-Tables: Specialize without information loss
  - based on relational table manipulations

What is a Side-Choosing Game Result?

- Combinatorial Game G
  - Not known whether there is a winning strategy
  - outcome: Proponent/Winner
- Side-Choice (P (Proponent) or O (Opponent))
- abbreviated: Side-Choice Result x Combinatorial Game Result

Sx (Side(x))	S!x (Side(!x))
P	O
O	P
P	P
O	O

P (Proponent)	Winner
x	x
x	!x
!x	x
!x	!x

One P against one O!

Result of a Side-Choosing Game Play

- A game that produces an SCG-Row in an SCG-table. First move is simultaneous: choose a side: Proponent (P) or Opponent (O).
- What is a row in an SCG-table?
  - Presents one game result between two distinct participants x and y.
  - Columns are: **W,L,F** (for **W**inner, **L**oser, **F**orced) (SCG-Table Rule 0).
  - **W**,**L** contain either x or y. **W≠L**. There are no ties (SCG-Table Rule 1).
  - **F** contains “none” or **W** or **L** (SCG-Table Rule 2).
    - A participant is Forced if it has to take the opposite side than it has chosen. Synonym: Devil’s Advocate.

Result of several Side-Choosing Game Plays (continued)

- What is an SCG-Table?
  - Given a set of Players.
  - A table of SCG rows satisfying rules 0-2 involving participants in Players.
  - Multiple rows may involve the same two players (SCG-Table Rule 3).
- To determine who wins the side war requires the execution of some protocol between the two participants. The SCG-Table definition does not specify the protocol: separation of concerns.

SCG-Tables

columns Concrete SCG-Table:
x: player
!x: other player
Sx: side of player x
S!x: side of player !x
P: Proponent

2/4

SideChoiceCases
core: **Sx, S!x**

2/4

SemanticGameCases
core: **W,P**

what we want

Abstract SCG-Table
core: **W, L, F**
derived: **Fault**, **Control**

3(5)/6

core columns abstract SCG-Table:
W: Winner
L: Loser
F: Forced

direct product

is-a

filter

drop: **Sx, S!x, P**

Concrete SCG-Table
core: **Sx, S!x, W, P**
derived: **F, L**

4(6)/12

derived columns SCG-Table:
Fault: Loser is not forced
Control: Loser in control

number of columns (with derived)/number of rows

Problem we solve next: Meritocracy Finding

- Given a tournament of side-choosing games among a set of Players, how can we find the most meritorious players?
- Note: There is a lot of noise produced by the tournament:
  - We don't know whether claim is true.
  - true claim might be refuted.
  - false claim might be defended.
  - players switch their sides between different games.
  - players may lie about their strength and lose intentionally to help a friend become more meritorious (collusion among players).
  - The more weak players or the more collusion, the more noise.
- HOW CAN WE FIND ORDER IN THIS COMPLEXITY?

Informal Reasoning

Case Abbreviation	Winner/Loser	Forced/Unforced
WF	Win	Forced
WU	Win	Unforced
LF	Lose	Forced
LU	Lose	Unforced

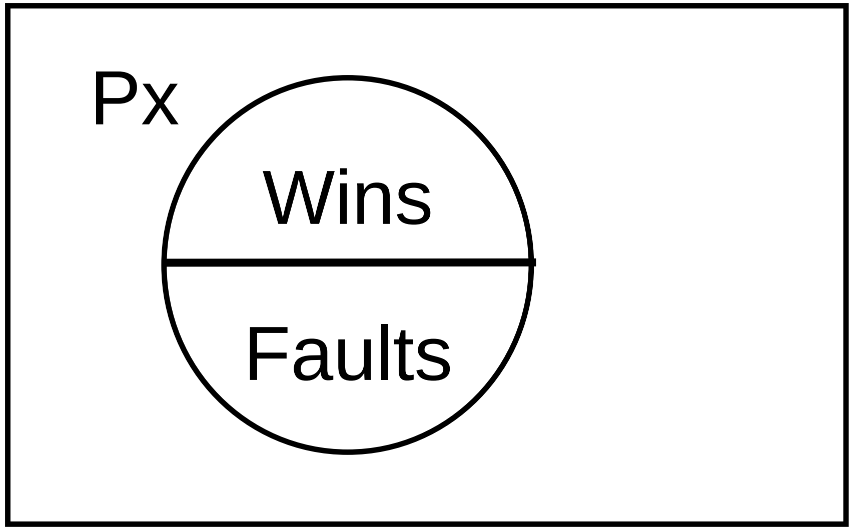
Which of the four statistics is a reliable indicator of strength or weakness?
WF: strength: no (because of possible collusion)
WU: strength: no (because of possible collusion)
LF: weakness: no (because of forcing)
LU (Fault): weakness: YES. Player is contradictory!
  Loser is Proponent: should have won
  Loser is Opponent: should have prevented the other from winning.

Informal argument why counting Faults is interesting.

Fair Peer-Based Evaluation for n participants

- Builds on Two-Participant Evaluation
- Ranking systems for side-choosing games
- Axiomatic treatment: collusion-resistant
  - Introduce 3 axioms

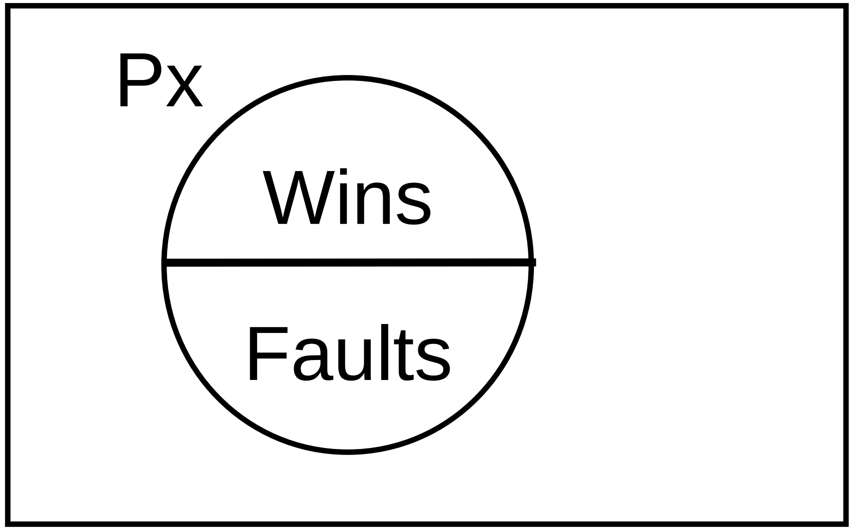
Non-Negative Effect For Wins (Axiom 1: NNEW)



Additional wins cannot worsen Px's rank w.r.t. other participants.

undisputed: Wins don't lower rank.

Non-Positive Effect For Losses (Axiom 2: NPEL)



Implies:

Additional losses cannot improve Px's rank w.r.t. other participants.

undisputed: Losses don't increase rank.

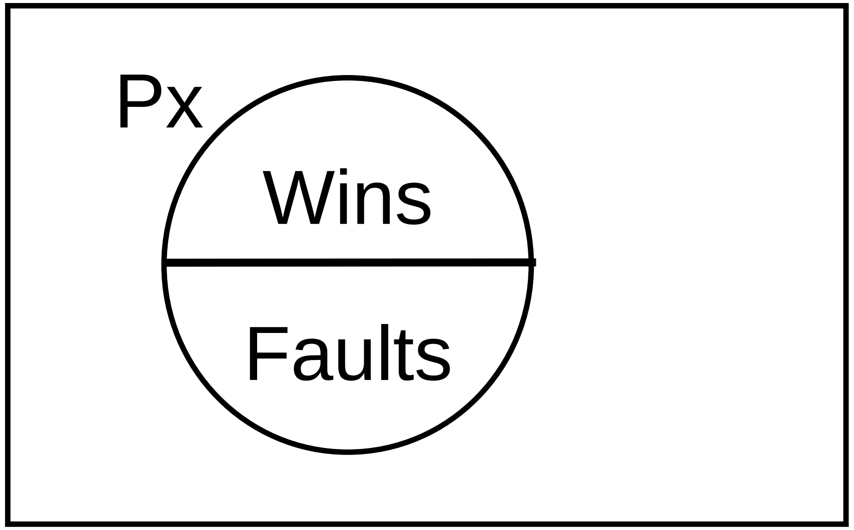
Ranking Functions (Anonymity)

- Output ranking is independent of participants' identities.
- Ranking function ignores participants' identities.
- Participants also ignore their opponents' identities.

Collusion-Resistance

- Slightly weaker notion than anonymity.
- What you want in practice.
- A participant Py can choose to lose on purpose against another participant Px, but that won’t make Px get ahead of any other participant Pz.

Collusion-Resistance (Axiom 3: CR)



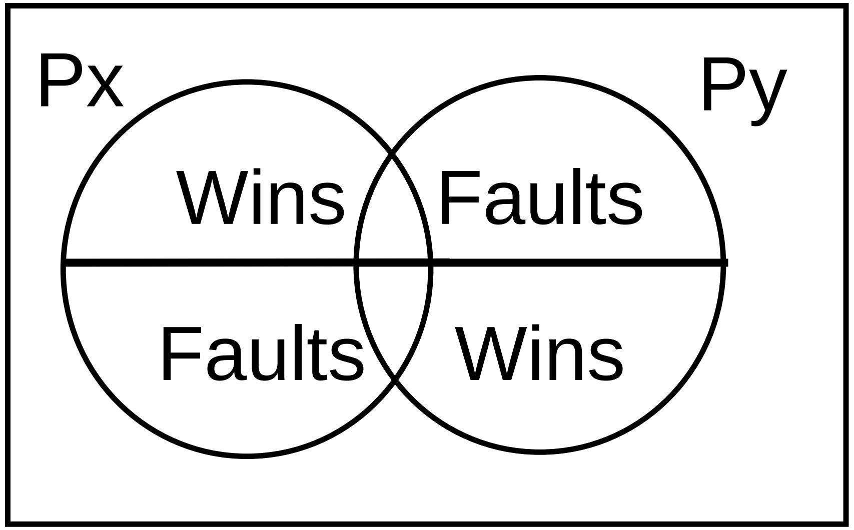
Games outside Px’s control cannot worsen Px’s rank w.r.t. other participants.

Px is in control if (Px in W(inner)) or (not(Px in F(orced)))

Collusion-Resistant Axiom: Explanation

- Your rank cannot come down when additional rows are added to table T
  - where you did not participate.
  - where you won.
  - where you were forced.

Preparing for the Discovery/Surprise: Locally Fault Based (LFB)



Relative rank of Px and Py depends only on faults made by either Px or Py.

“Locally” is used to exclude faults made by some third player z when ranking x versus y.

Discovery: Property of SCG-Tables

- A useful design principle for ranking functions.
- Under NNEW, NPEL : CR = LFB
- LFB is quite unusual.
- LFB lends itself to implementation.
- Not only is fault counting important; it is fundamental.

NNEW $\wedge$ CR => LFB
NPEL $\wedge$ LFB => CR
Representation Theorem: NNEW $\wedge$ NPEL => (CR $\Leftrightarrow$ LFB)

Proof
Venn Diagram for Game Kinds involving x,y

WF(x), WF(y),LF(x),LF(y) shown in diagram
1∪3-WF(x)=WU(x); 4∪6-WF(y)=WU(y); 2∪4:LU(x); 3∪5:LU(y)

≥ monotonically non-decreasing
≤ monotonically non-increasing

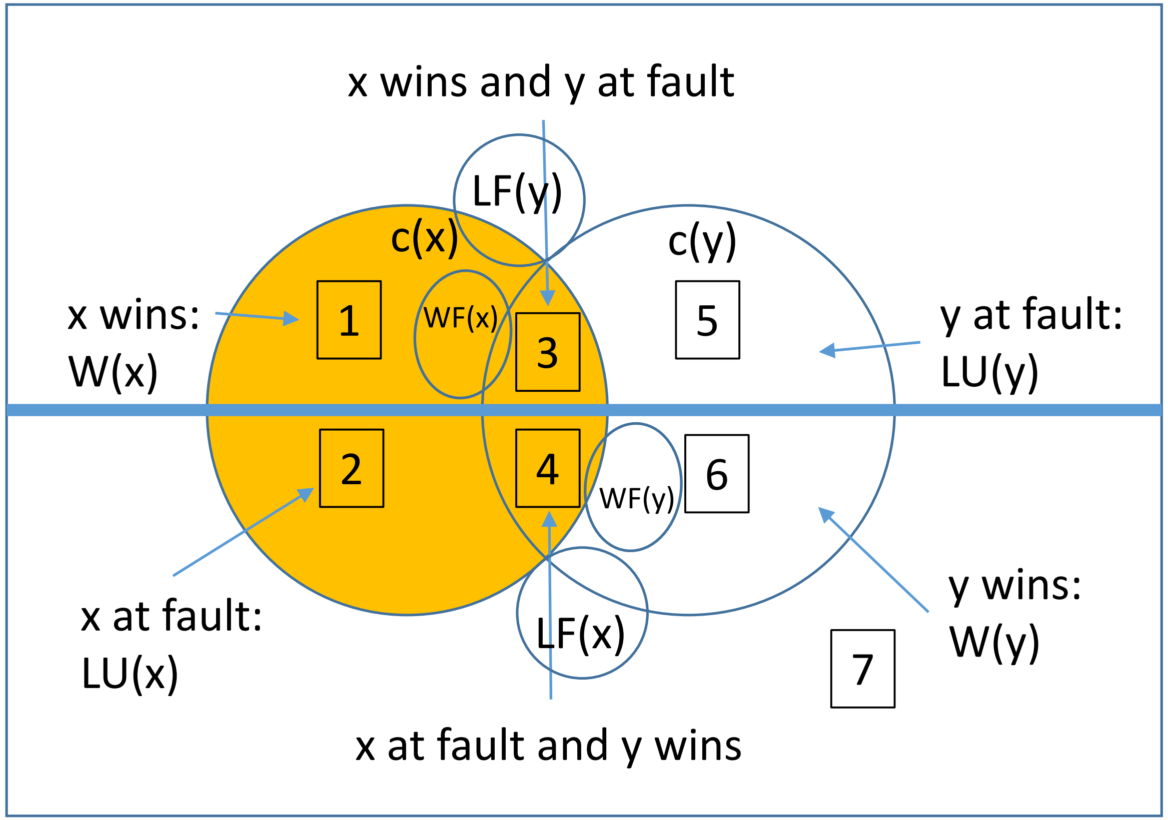
LFB: $d_W U = 0$, $d_{LF} U = 0$

All games

games that cannot improve rank of x wrt. y:
6 by NNEW
1,7 by CR

games that cannot worsen rank of x wrt. y:
6,7 by CR
1 by NNEW

Proves:
NNEW and CR => LFB



2, 3, 4, 5: LFB

1 [!fl y] [w x] = [!c y] [w x]
2 [!w y] [fl x] = [!c y] [fl x]
3 [fl y] [w x] = [c y] [w x]
4 [w y] [fl x] = [c y] [fl x]
5 [!w x] [fl y] = [!c x] [fl y]
6 [!fl x] [w y] = [!c x] [w y]
7 [!c x] [!c y]

c(z) = z in control (full circle)
= (z wins) or (z not forced)

NNEW: $d_W U \leq 0$, NPEL: $d_L U \geq 0$, CR: $d_W U \geq 0$, $d_{LF} U = 0$, $d_{LU} U \geq 0$

Ranking Axioms Imply

≥ monotonically non-decreasing
≤ monotonically non-increasing

Monotonicity Constraints

- NNEW: $d_{WF}U \leq 0 \wedge d_{WU}U \leq 0$
- NPEL: $d_{LF}U \geq 0 \wedge d_{LU}U \geq 0$
- CR: $d_{WF}U \geq 0 \wedge d_{WU}U \geq 0 \wedge d_{LF}U = 0 \wedge d_{LU}U \geq 0$
- LFB: $d_{WF}U = 0 \wedge d_{WU}U = 0 \wedge d_{LF}U = 0$

Above implies:
NNEW Λ CR => LFB
NPEL Λ LFB => CR
NNEW Λ NPEL => (CR ⇔ LFB)

Finding a concrete LFB function.

- One example of an infinite family.

Fault-Counting Scoring Function

- Provide a concrete example of a ranking function which is NNEW, NPEL and LFB and therefore CR.
  - Players are ranked according to their score: the number of faults they make. The fewer the number of faults the higher the rank.
- Satisfies the NNEW and NPEL
  - Clearly, Fault Counting is LFB and therefore CR (by previous theorem).
- Note: There is an infinite family of ranking functions that are LFB.

Incentive Compatibility Theorem

- We call it “Incentive Compatibility Theorem” (for side-choosing games)
  - it shows that without collusion-resistance (CR) incentive compatibility is not guaranteed.
- Incentive compatibility means
  - the best players will be top ranked.
  - “best” means: quasi-perfect. Might be wrong but nobody can illustrate through game play.

Incentive Compatibility Theorem

- A quasi-perfect player is a player who never makes a fault.
- (A perfect player always chooses the correct side and always wins the defense and therefore is quasi-perfect).
- Meritocracy Theorem
  - Part 1: [CR is necessary]: If ranking function is not collusion-resistant (CR) but NNEW and NPEL, there exists a set of games where a quasi-perfect player is not top-ranked.
  - Part 2: [CR is sufficient] NNEW and NPEL and CR imply that quasi-perfect player is top-ranked.

Quasi-Perfect Player

- The concept of a quasi-perfect player is an important one. It indicates that the outcome of side-choosing games depends on the quality of the players.
- It is possible that a false claim is regularly defended. But the other players don’t have the skill to refute the claim.
- Being a quasi-perfect player is already an accomplishment, even if the claim is false. It shows superior skill with respect to the other players.

Summary Theory

- Side-choosing games require a side-choice at the beginning and then resort to forcing if both choose same side. A side-choosing game finishes with one or more combinatorial games.
- The axiomatic ranking theory producing the representation theorem: NNEW Λ NPEL => (CR ⇔ LFB), only depends of SCG-Tables satisfying rules 0-3 and not on the details of combinatorial games. The 4 rules provide an abstraction barrier from the details of protocols.
- CR is crucial for incentive compatibility.

Application Concepts

Lab = (Side-Choosing Game, Participants)

Participant

Avatar/
Software Agent

Competition System we built
Baby Avatar automatically generated from Side-Choosing Game

Human

Tools like Piazza are sufficient but could be improved

Methods

- Use piazza.com
  - Use JSON for scientific discourse (objects sent back and forth during combinatorial game).
  - Divide class into teams of 3.
- For strategies in software
  - We developed
    - a generator for baby strategies
    - administrator automated
- Also used simulation (synthetic strategies) to help invent the theory.

Applications of SCG (1)

- Teaching Software Development
  - Students get Baby-Strategy to start (automatically generated from claim).
  - Students add intelligence to their strategy to outperform the strategies of their peers.
  - A strategy consists of a function for side choice and a function for each quantifier (we use simplified semantic games).
  - Strategy with fewest losses wins. Strategies use web for fight!
  - Weekly tournaments with slightly modified claim: encourage well-modularized software that is easy to change.

Applications of SCG (2)

- Bring order to literature on solving computational problems.
  - Authors are required to provide a strategy to defend the claim.
  - Knowledge becomes active on the web.
  - A newcomer might beat all current strategies. Easily verified by a tournament.
  - Develop a Wikipedia for computational problems.

Applications of SCG (3)

- Teaching Formal Sciences (e.g., Algorithms).
  - Use a platform like Piazza to execute the protocol.
  - Avoid claims where the semantic game gives away the solution.
  - Divide class into groups of size 3. Balance skills in each group. Play tournaments within groups to prevent information overload for class.
  - Students came up with solutions that were about 10 years behind the state-of-the-art.

Price for peer evaluation

- Given: A side-choosing game s.
  - Find winning strategy (or an approximation to it) for
    - winning the game s and
    - exposing weaknesses (if they exist) in any strategy for defending the claim.
      - For second bullet: You play in devils advocate role and still try to win against a non-perfect player.
- The symmetric requirement has three benefits
  - fair peer evaluation
  - “dynamic benchmark” instead of a static benchmark.
  - requires the players to think about the difficult cases.

Results

- SCG usage for teaching using forum
  - Innovation Success with Undergraduates using SCG on piazza.com: Qualitative Data Sources & Analysis
  - Perfect for creating interaction between students (peer teaching)
- Strategy competitions are useful for teaching (and good for competitive innovation).
  - Some students stayed up to see their strategy succeed in the tournament.

Related Work

- Semantic Games
- Rating and Ranking Functions
- Tournament Scheduling
- Match-Level Neutrality

Rating and Ranking Functions (I)

- Dominated by heuristic approaches
  - Elo ratings.
  - Who’s #1?
- There are axiomatizations of rating functions in the field of Paired Comparison Analysis.
  - CR not on radar

Rating and Ranking Functions (II)

- Rubinstein[1980]:
  - points system (winner gets a point) characterized as:
    - Anonymity : ranks are independent of the names of participants.
    - Positive responsiveness to the winning relation which means that changing the results of a participant p from a loss to a win, guarantees that p's rank would improve.
    - IIM: relative ranking of two participants is independent of matches in which neither is involved.
  - "beating functions" are restricted to complete, asymmetric relations.

Tournament Scheduling

- Neutrality is off radar.
  - Maximizing winning chances for certain players.
  - Delayed confrontation.

Match-Level Neutrality

- Dominated by heuristic approaches
  - Compensation points.
  - Pie rule.

Future Work

- Problem decomposition labs.
- Social Computing.
- Evaluating Thoroughness.

Financial Support

- Novartis

Conclusions

- Side-choosing games provide a useful foundation to organize computational problem solving and formal science communities for research and experience-based learning.
- We found a solution to the problem of lowering the barrier of entry for competition organizers by shifting evaluation tasks from the administrator to the players.
- We show the fundamental nature of locally fault-based evaluation in the presence of collusion-resistance.
- Simple result: 3 axioms, representation theorem, adapted full round-robin tournament with fault-counting gives fair, collusion-resistant evaluation of SCGs.

CCIS

- 45 faculty + 9 researchers
  - 7 new faculty in 2012
  - 6 new faculty in 2013
- 634 undergrads, 606 MS, 97 PhDs
- $7.6 million in grant funding in 2012
- PhD Programs in:
  - Computer Science
  - Information Assurance
  - Health Informatics
  - Network Science

Research at CCIS

- Security
  - 19+ papers at *NDSS*, *CSS*, *S&P*, and *Usenix Security* since '08
  - Funding from DARPA, Symantec, and Verisign
- Programming Languages
  - 19 papers in *POPL*, *OOPSLA*, and *ICFP* since '08
  - Active in the ECMA Javascript standards body
- DB, IR, and ML
  - Papers in *SIGIR*, *CIKM*, *KDD*, *SIGMOD*, *VLDB*, *ICDM*
  - New research center for digital humanities
- Formal Methods
  - 10 papers in *FMSD*, FMCAD, *TOPLAS*, and *CAV* since '08
  - NSF CAREER for building dependable concurrent software
- Robotics and Computer Vision
  - NSF CAREER for building robots that handle uncertainty
  - New hire for 2013: Rob Platt
- Network Science
  - Founders of the field: Laszlo Barabasi and Alex Vespignani
  - 20+ papers in *Nature*, *Science*, and *PNAS* since '08

Questions?

Karl Lieberherr
lieber@ccs.neu.edu

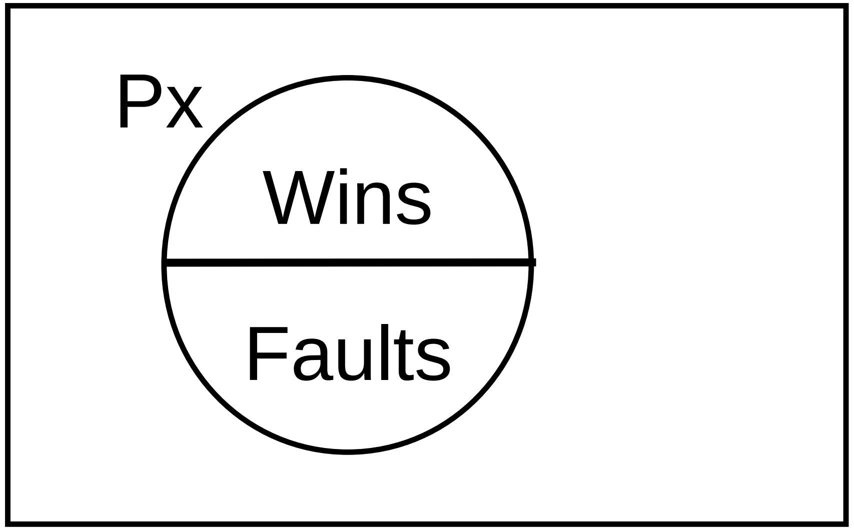
Thank You!

For Hansruedi

- Proper tournament evaluation for side-choosing games.
- Games with winners and losers, no draws.
- Evaluation must satisfy 3 axioms.
- Collusion-resistant axiom: Games outside a player's control cannot lower that player's rank with respect to other players.

Collusion-Resistance (Axiom 3: CR)

Px is not in control if not (Px in W(inner)) and (Px in F(orced))
Px is not in control if Px does not participate or (Px in L(oser) and (Px in F(orced))



Games outside Px's control cannot worsen Px's rank w.r.t. other participants.

Px is in control if (Px in W(inner)) or (not(Px in F(orced)))
Fault: Loser is not forced.

Side-Choosing Game

- Combinatorial game G
- Proponent: start position is a winning position.
- Opponent: start position is not a winning position.
- Choose side.
- Need for forcing.
- Faults.

end Hansruedi

Combinatorial Games

Examples of Combinatorial Games

- Truth of sentences in various logics
  - First-order predicate logic
  - Second-order predicate logic
  - etc.
- Positions in 2-person extensive-form games with perfect information. Choose a node (position) and ask: is it winning? Example: Mate in 2.
- From Combinatorial Game Theory: chess, checkers, Nim (solved), etc.

Standard Example: SCG Rules

fal = Opponent
ver = Proponent

Φ	Move	Subgame
$\forall x : \Psi(x)$	fal provides x_0	$SG(\langle \Psi[x_0/x], A\rangle, ver, fal)$
$\Psi \wedge \chi$	fal chooses $\theta \in \{\Psi, \chi\}$	$SG(\langle \theta, A\rangle, ver, fal)$
$\exists x : \Psi(x)$	ver provides x_0	$SG(\langle \Psi[x_0/x], A\rangle, ver, fal)$
$\Psi \vee \chi$	ver chooses $\theta \in \{\Psi, \chi\}$	$SG(\langle \theta, A\rangle, ver, fal)$
$\neg\Psi$	N/A	$SG(\langle \Psi, A\rangle, fal, ver)$
$p(x_0)$	N/A	N/A

Alt. Comb. Game: Perfect-Information Game in Extensive Form (from Shoham & Leyton-Brown)

- A finite perfect-information game in extensive form is a tuple G=(N,A,H,Z,χ, ρ, σ,u), where
  - N is a set of n players;
  - A is a set of actions;
  - H is a set of non-terminal choice nodes;
  - χ: H -> 2^A is the action function, which assigns to each choice node a set of possible actions;
  - ρ: H -> N is the player function, which assigns to each nonterminal node a player i in N who choses an action at that node;
  - σ: H x A -> H ∪ Z is the successor function, which maps a choice node and an action to a new choice node or terminal node such that for all h1,h2 in H and a1,a2 in A, if σ(h1,a1)=σ(h2,a2) then h1=h2 and a1=a2; and
  - u=(u_1,…,u_n), where u_i:Z->R is a real-valued utility function for player i on terminal nodes Z.

Side-Choosing Games

- Benefits
- Based on Combinatorial Games
- Side-Choosing Game = (Combinatorial Game, Agreement Map)
  - side-choice followed by sequential game

benefits: Objective, Low Overhead, Correct, Targeted Feedback, Well-Defined Interfaces.

Benefits of Side-Choosing Games

- Objective: The result depends on how well the participants solve the computational problems coming from the claim/game.
- Low Overhead on Administrator: Prepare claim/game and check that the game rules are followed.
- Correct: The winners demonstrate their opponent's lack of skills for current claim/game.
- Targeted Feedback: game plays give losers specific feedback.
- Participants interact through well-defined interfaces. Choosing side and following game protocol.

Prominent Example of a Side-Choosing Game

- The semantic game for some logic, e.g., predicate logic, is a combinatorial game. The start position is a sentence ϕ interpreted in some structure A.
- Turn it into a side-choosing game by selecting an agreement map.

Toy Example: SCG Trace

✔ Proponent
✘ Opponent

SCG(∀x ∈ [0,1]: ∃ y ∈ [0,1]: x + y > 1.5, ✔, ✘)

✘ Provides 1 for x | Weakening (too much!)

SCG(∃ y ∈ [0,1]: 1 + y > 1.5, ✔, ✘)

✔ Provides 1 for y | Strengthening

SCG(1 + 1 > 1.5, ✔, ✘) ✔ Wins

SCG = Side-Choosing Game
SCG = Scientific Community Game
SCG = Specker Challenge Game
SCG = SemantiC Game
all make are meaningful

Abstraction Barriers/Important Steps

- Widen: Semantic Games – Combinatorial Games: Generalize
  - Simplifies Meritocracy Theory for Side-Choosing Games
- Minimum Recording: Side-Choosing Game Plays – SCG-Tables: Specialize without information loss
  - based on relational table manipulations
- Meritocracy: SCG-Tables – Player Ranking: Specialize
  - based on axioms

What is a Side-Choosing Game Result?

- Combinatorial Game G
  - Not known whether there is a winning strategy
  - outcome: Proponent/Winner
- Side-Choice (P (Proponent) or O (Opponent))
- abbreviated: Side-Choice Result x Combinatorial Game Result

Sx (Side(x))	S!x (Side(!x))
P	O
O	P
P	P
O	O

P (Proponent)	Winner
x	x
x	!x
!x	x
!x	!x

One P against one O!

Side-Choosing Games, Overview

- Motivated by semantic games. Sentence ϕ for structure A maps to game G((ϕ,A), proponent, opponent).
- Replace semantic game by combinatorial game without draw to abstract from logical formalism behind semantic games. Generalize to simplify meritocracy theory (application of Polya's Inventor's Paradox).
- A side-choosing game is a combinatorial game with predictive information but is not necessarily again a combinatorial game (randomness maybe in side-choosing game).

Agreement Map

- The agreement function maps two players x,y with their design-time side choices dx=dy for combinatorial game G into a set of plays between the two players with run-time side choices rx, ry such that at most one player is forced.
- We choose the following agreement function CAG (Competitive Agreement Game):
  - Randomly choose z, one of the players x,y and force z.
    - Play SCG(G, x, y, z, dx, dy, rx, ry)
    - Play SCG(G, x, y, !z, dx, dy,!rx, !ry)
- Play SCG(G, x, y, z, dx, dy, rx, ry) is a game between x and y, where z is forced. If x is forced then rx=!dx and ry=dy. If y is forced then rx=dx and ry=!dy. The claim is: in the start position of G the player in the P role (d*=P) has a winning strategy.
- dx,dy,rx,ry have values in {P,O}. z has values in {x,y}. !z =x if z=y. !z = y if z=x. !P=O, !O=P.

Motivation for CAG

- For some claims it is a disadvantage to have to move first because you give away a “secret” (e.g., the chess example Mate-in-2). Therefore, we choose the forced player randomly to balance the potential disadvantage.
- We play two games to give each player a chance to test the other.

Alternative for CAG

- CAG-1/2 only plays one game choosing the forced player randomly.
- We choose the following agreement function CAG-1/2 (Competitive Agreement Game 1/2):
  - Randomly choose z, one of the players x,y and force z.
    - Play SCG(G, x, y, z, dx, dy, rx,   ry)
  - Play SCG(G, x, y, z, rx, ry) is a game between x and y, where z is forced. If x is forced then rx=!dx and ry=dy. If y is forced then rx=dx and ry=!dy.

What is a Side-Choosing Game (G,AM) for a Combinatorial Game G and an Agreement Map AM?

1. (G,AM) is played by x and y.
2. Game
   1. Choose a design-time side: P or O: dx, dy for G. P means: The start position of G is a winning position for player in P role. O means: The start position is not a winning position for player in P role.
   2. If dx!=dy: play game SCG(G, x, y, dx, dy, dx, dy)
   3. Else: Use agreement map AM to determine games to be played. Each game is of the form SCG(G, x, y, dx, dy, rx, ry) for run-time side-choices rx, ry determined by the agreement map AM.
   4. Each game result is recorded as a triple: W,L,F.

From Combinatorial to Side-Choosing Game

- CG = Combinatorial Game, no notion of proponent, opponent.
- SCG = Side-Choosing Game, notion of proponent/opponent at design-time and run-time.
  - design-time proponent/opponent (choice by players)
    - Proponent claims: start position of CG G is winning position for player who is run-time proponent.
  - run-time proponent/opponent (influenced from outside, AM)
- Same Information
  - CG(G, x, y, dx, dy, rx, ry)
  - CG(G, x, y, z, rx, ry), z is forced player.
- result (W,L,F). W,L,F have values in {x,y}.
- A side-choosing game is a combinatorial game with predictive information but is not necessarily again a combinatorial game (randomness in AM).
- A side-choosing game is a generalization of a semantic game. Semantic games also use the notion of proponent/opponent but are tied to a logical formalism which is irrelevant for studying meritocracy management in side-choosing games.

Result of a Side-Choosing Game Play

- A game that produces an SCG-Row in an SCG-table. First move is simultaneous: choose a side: Proponent (P) or Opponent (O).
- What is a row in an SCG-table?
  - Presents one game result between two distinct participants p and q.
  - Columns are: **W,L,F** (for **W**inner, **L**oser, **F**orced) (SCG-Table Rule 0).
  - **W**,**L** contain either p or q. **W≠L**. There are no ties (SCG-Table Rule 1).
  - **F** contains “none” or **W** or **L** (SCG-Table Rule 2).
    - A participant is Forced if it has to take the opposite side than it has chosen. Synonym: Devil’s Advocate.

Result of several Side-Choosing Game Plays (continued)

- What is an SCG-Table?
  - Given a set of Players.
  - A table of SCG rows satisfying rules 0-2 involving participants in Players.
  - Multiple rows may involve the same two players (SCG-Table Rule 3).
- To determine who wins the side war requires the execution of some protocol between the two participants. The SCG-Table definition does not specify the protocol: separation of concerns.

SCG-Tables

columns Concrete SCG-Table:
x: player
!x: other player
Sx: side of player x
S!x: side of player !x
P: Proponent

2/4

SideChoiceCases
core: **Sx, S!x**

2/4

SemanticGameCases
core: **W,P**

what we want

Abstract SCG-Table
core: **W, L, F**
derived: **Fault**, **Control**

3(5)/6

core columns abstract SCG-Table:
W: Winner
L: Loser
F: Forced

direct product

is-a

filter

drop: **Sx, S!x, P**

Concrete SCG-Table
core: **Sx, S!x, W, P**
derived: **F, L**

4(6)/12

derived columns SCG-Table:
Fault: Loser is not forced
Control: Loser in control

number of columns (with derived)/number of rows

Core sets, Dan Feldman

- Given data D and Algorithm A with $A(D)$ intractable, can we efficiently reduce D to C so that $A(C)$ fast and $A(C) \sim A(D)$?

Illustrate that collusion-resistance is essential

- best player not on top if wins against non-forced are counted.
- A,B,C,D players. D perfect, B,C colluding, A higher ranked than D.
  - all non-forced losses?
  - A-B 1:0 collusion: x game plays
  - A-C 1:0 collusion
  - A-D 0:1
  - B-C 0:1
  - B-D 0:1
  - A x+1 wins, B 0 wins, C 1 win, D 1 win: A overall winner
  - A 1 fault, B x+2 faults, C 2 faults, D 0 faults: D overall winner
  - engage D again: C-D 0:1: still A winner for win counts

Claim: In full round-robin tournament

- can count wins and still have perfect on top?
- Would contradict the representation theorem!

- when is the perfect on top?
- connection to collusion-resistance?
- perfect has 0 faults.

- in full-round robin tournament for n players:
  - perfect wins all n-1 games, 0 faults
  - others can artificially play more games and win more than n-1 but no longer full-round-robin tournament.

Table extension

- axioms are about table extensions
- full round-robin has a fixed number of games, axioms not applicable

Incentive Compatibility Theorem

- We call it “Incentive Compatibility Theorem” (for side-choosing games)
  - it shows that without collusion-resistance (CR) incentive compatibility is not guaranteed.
  - with CR, incentive compatibility is guaranteed.
- Incentive compatibility means
  - the best players will be top ranked.
  - “best” means: quasi-perfect. Might be wrong but nobody can illustrate through game play.

Incentive Compatibility Theorem

- A quasi-perfect player is a player who never makes a fault.
- (A perfect player always chooses the correct side and always wins the defense and therefore is quasi-perfect).
- Meritocracy Theorem
  - Part 1: [CR is necessary]: If ranking function is not collusion-resistant (CR) but NNEW and NPEL, there exists a set of games where a quasi-perfect player is not top-ranked.
  - Part 2: [CR is sufficient] NNEW and NPEL and CR imply that quasi-perfect player is top-ranked.

Quasi-Perfect Player

- The concept of a quasi-perfect player is an important one. It indicates that the outcome of side-choosing games depends on the quality of the players.
- It is possible that a false claim is regularly defended. But the other players don’t have the skill to refute the claim.
- Being a quasi-perfect player is already an accomplishment, even if the claim is false. It shows superior skill with respect to the other players.

- A semi-perfect player is a player who never makes a fault.
- A perfect player is a player who always chooses the correct side and properly defends it.
- Fact: A perfect player is semi-perfect.
- A semi-perfect player looks like a perfect player but the claim it defends could actually be refuted. (either claim is false or defense is not optimal)
- A player is top-ranked if it never makes a fault.
- Theorem: If collusion-resistance is violated, a perfect or semi-perfect player may not be top-ranked.

better: collusion-resistance is essential

- A quasi-perfect proponent is a player who never makes a fault after choosing the proponent role.
- A quasi-perfect opponent is a player who never makes a fault after choosing the opponent role.

- A quasi-perfect player is a player who never makes a fault.
- A perfect player is a player who always chooses the correct side and properly defends it.
- Fact: A perfect player is quasi-perfect.
- A quasi-perfect player looks like a perfect player but the claim it defends could actually be refuted. (either claim is false or defense is not optimal)
- Theorem [CR is essential]: If ranking function is not collusion-resistant (CR) but NNEW and NPEL, there exists a set of games where a quasi-perfect player is not top-ranked.
- The theorem means that the optimal solution might not win. This would be an absolute show stopper. The truth would be distorted!
- Proof: Give an example of a tournament where a semi-perfect player is not top ranked.
- What is the best example of a tournament to prove the theorem?
- Theorem: If ranking function is NNEW, NPEL and CR then for all sets of games all quasi-perfect players are top-ranked.

Robust Ranking Functions for Side-Choosing Games

- player is never lower ranked than its final rank
  - corollary: quasi-perfect player will stay among top-ranked
- If NNEW, NPEL and CR then ranking function is robust.
- Definition: A ranking function is robust if for all sub tournaments S of a tournament T all players are ranked higher or equal for S than for T.
- f: table -> ranking (total preorder)

Theorems to present in paper/talks

- Representation Theorem: NNEW and NPEL => (CR⇔LFB)
- "CR is essential": without CR, quasi-perfect player may not be top-ranked.
  - Meritocracy Theorem: NNEW and NPEL and CR imply that quasi-perfect player is top-ranked.
  - NNEW and NPEL and CR imply that the ranking function is robust.
- Definition: A ranking function is robust if for all sub games S of a set of games T all players are ranked higher or equal for S than for T.
- Meritocracy Theorem
  - Part 1: [CR is necessary]: If ranking function is not collusion-resistant (CR) but NNEW and NPEL, there exists a set of games where a quasi-perfect player is not top-ranked.
  - Part 2: [CR is sufficient] NNEW and NPEL and CR imply that quasi-perfect player is top-ranked.

Quasi-Perfect Player

- The concept of a quasi-perfect player is an important one. It indicates that the outcome of side-choosing games depends on the quality of the players.
- It is possible that a false claim is regularly defended. But the other players don't have the skill to refute the claim.
- Being a quasi-perfect player is already an accomplishment, even if the claim is false. It shows superior skill with respect to the other players.

Equilibrium

- When 2 players x and y always win when they are not forced, then the game is in an equilibrium. This can be viewed as a tie or draw between x and y.

Interpreted logical formulas

- The structure A must be completely specified and might satisfy a certain set of axioms.
- The logical formula must be complete and might involve a set of axioms that must hold.

Agreement Algorithm

- The agreement algorithm maps two players x,y with their design-time side choices dx=dy for combinatorial game G into a set of plays between the two players with run-time side choices rx, ry such that at most one player is forced.
- We choose the following agreement algorithm CAA (Competitive Agreement Algorithm):
  - Randomly choose z, one of the players x,y and force z.
    - Play SCG(G, x, y, z, dx, dy, rx, ry)
    - Play SCG(G, x, y, !z, dx, dy,!rx, !ry)
- Play SCG(G, x, y, z, dx, dy, rx, ry) is a game between x and y, where z is forced. If x is forced then rx=!dx and ry=dy. If y is forced then rx=dx and ry=!dy. The claim is: in the start position of G the player in the P role (d*=P) has a winning strategy.
- dx,dy,rx,ry have values in {P,O}. z has values in {x,y}. !z =x if z=y. !z = y if z=x. !P=O, !O=P.

Alternative for CAG

- CAG-1/2 only plays one game choosing the forced player randomly.
- We choose the following agreement function CAG-1/2 (Competitive Agreement Game 1/2):
  - Randomly choose z, one of the players x,y and force it.
    - Play SCG(G, x, y, z, dx, dy, rx, ry)
  - Play SCG(G, x, y, z, dx, dy, rx, ry) is a game between x and y where z is forced. If rx=!dx and ry=dy then x is forced. If rx=dx and ry=!dy then y is forced.

- Side-choosing games using social choice theory terminology
- From SEP

- Aggregation rule
  - set of individuals N = {1,2, ... ,n}, n>=2.
  - combinatorial game with starting position p(i,j).
- choose two alternatives di,dj
- design-time choices
- di = proponent: starting position p(i,j) winning for i
- dj = opponent: starting position p(i,j) not winning for i

Definition: Side-Choosing Game

- A side-choosing game is a triple (G,SC,AA), where G is a two-person, draw-free, combinatorial game, SC is a side-choice configuration and AA is an agreement algorithm.
  - SC: We have two players, x and y, who make a choice dx and dy for the start position of G. dx, dy are elements of {P,O}. If x chooses dx=P then x claims to win the combinatorial game G from its start position. If x chooses dx=O then x claims to prevent y from winning G from its start position. The players make the side choices but SC specifies the configuration. Examples for SC:
    - simultaneous
    - sequential
      - with probability p for x to be the first player.
      - other context-sensitive mechanisms to choose the first player.
  - AA:

Side-choosing Game (continued)

Meritocracy Theorem (old)

- Meritocracy Theorem: NNEW and NPEL and CR imply that quasi-perfect player is top-ranked.
- CR is essential if we don't have control about the games played.
- A full round-robin tournament also guarantees that a quasi-perfect player is top-ranked.
- The meritocracy theorem guarantees the property for any set of games.
- A quasi-perfect player can be labeled as a good player. It defends its side-choice each time.

Definition: Side-Choosing Game

- A side-choosing game is a triple (G,SC,AA), where G is a two-person, draw-free, combinatorial game, SC is a side-choice configuration and AA is an agreement algorithm.
- G, SC and AA are defined separately. The important component is the combinatorial game G; SC and AA offer variation possibilities to define the side-choosing games. We will use simple instances of SC and AA for our side-choosing games.

Combinatorial Game Definition

1. There are two players. Sequential (turn-based). Perfect-Information, no chance or hidden information. (But players may hide their winning strategies.)
2. There is a finite set of possible positions of the game.
3. The rules of the game specify for both players and each position which moves to other positions are legal moves.
4. The game ends when a position is reached from which no moves are possible. A predicate on the final position determines who has won. There is an absolute winner: the first player to fulfill the winning condition. No ties or draws.
5. The game ends in a finite number of moves.

Side-Choice Configuration Definition

- The side choices are made by the players but the Side-choice Configuration (SC) defines the sequencing of the design-time decisions.
- We have two players, x and y, who make a choice dx and dy for the start position of the combinatorial game G. dx, dy are elements of {P,O}. If x chooses dx=P then x claims to win the combinatorial game G from its start position. If x chooses dx=O then x claims to prevent y from winning G from its start position. The players make the side choices but SC specifies the configuration. Examples for SC:
    - simultaneous
    - sequential
        - with probability p for x to be the first player.
        - other context-sensitive mechanisms to choose the first player.

Agreement Algorithm Definition

- The agreement algorithm maps two players x,y with their design-time side choices dx=dy for combinatorial game G into a set of plays between the two players with, for each play, run-time side choices rx, ry such that at most one player is forced.
  - Example: Agreement algorithm CAA (Competitive Agreement Algorithm):
    - Randomly choose z, one of the players x,y and force z.
      - Play SCG(G, x, y, dz, d!z, !dz, d!z); z is forced
      - Play SCG(G, x, y, dz, d!z, dz, !d!z); !z is forced
    - Play SCG(G, x, y, dx, dy, rx, ry) is a game between x and y, where x makes design-time choice dx and y makes design-time choice dy. The run-time choices are rx and ry. If x is forced then rx=!dx and ry=dy. If y is forced then rx=dx and ry=!dy. The claim is: in the start position of G the player in the P role (d*=P) has a winning strategy.
    - dx,dy,rx,ry have values in {P,O}. z has values in {x,y}. !z =x if z=y. !z = y if z=x. !P=O, !O=P.

Motivation for CAA

- For some claims it is a disadvantage to have to move first because you give away a “secret” (e.g., the chess example Mate-in-2). Therefore, we choose the forced player randomly to balance the potential disadvantage.
- We play two games to give each player a chance to test the other.

Alternative Example for Agreement Algorithm

- CAA-1/2 only plays one game choosing the forced player randomly.
- CAA-1/2 (Competitive Agreement Algorithm 1/2):
  - Randomly choose z, one of the players x,y and force z.
    - Play SCG(G, x, y, dz, d!z, !dz, d!z); z is forced

Difference: Balanced Approach

- Not only Skolem functions SE for Existential quantifiers of ϕ.
- Also Skolem functions SA for $!\phi$.
- Skolem functions SA for $!\phi$ are supposed to detect any weaknesses in the Skolem functions SE.
- Special case: inputs that break algorithm (produce wrong result).

Price for peer evaluation

- Given: A claim (A, ϕ).
- Find perfect solution (or an approximation to it) for
  - defending the claim and
  - exposing weaknesses (if they exist) in any defense of the claim.
- Given: A side-choosing game s.
- Find winning strategy (or an approximation to it) for
  - winning the game s and
  - exposing weaknesses (if they exist) in any strategy for defending the claim.
- For second bullet: You play in devils advocate role and still try to win against a non-perfect player.

Yaron Singer's Course: CS 284r: Incentives and Information in Networks

- Premise: Human Interactions follow mathematical patterns that can be leveraged by algorithms for acquiring. disseminating, and learning information.

Connection to Yaron's Premise

- We study engineered networks where participants need to follow a protocol of interaction about claims.
- We study algorithms that take the results of those constrained interactions as input and produce
  - information about the participants and, indirectly through the participants,
  - information about the claims.
- We study techniques to engineer networks that disseminate specific knowledge through a network.

Connection to Yaron’s Work

- So while Yaron gathers information from networks "in the wild" we gather information from networks we have constrained by rules. Our rules are motivated by observing how formal scientific communities work.

How does Merlin convince Arthur that the Claim C is true

- Merlin gives a proof of C in a proof system. Arthur checks.
- Merlin provides a winning strategy for the semantic game associated with the claim C and proves it correct in a proof system. Arthur checks.
- Merlin consistently wins the semantic game associated with the claim C against Arthur. Arthur agrees with Merlin after failing to refute the claim after multiple attempts.
  - Now Arthur orders Merlin to give out his strategy and Arthur can defend the claim against other Kings, without having a proof that it works.

Truth is secondary to construction

- It could be that Arthur cannot find the hole in the reasoning of Merlin but actually, the claim is false.

From Semantic Games to SCG