SCG Court: A Crowdsourcing Platform for Scientific Innovation using Unreliable Scientists

Ahmed Abdelmeged
Northeastern University
mohsen@ccs.neu.com

Karl Lieberherr
Northeastern University
lieber@ccs.neu.edu

ABSTRACT

We apply the Scientific Community Game (SCG, formerly called the Specker Challenge Game) to crowdsourcing scientific innovation. SCG is the first generic model of the Popperian Scientific Method on the web and has several applications to improve crowdsourcing on the web.

We use the SCG Design Pattern(SGDP) to study variations of SCG that are useful for learning.

SCG is designed to be both educational for scholars, and to solve problems that the students don't know how to solve yet.

Broaden paper: Focus on Call for Papers item: Knowledge, education, and scholarship on and through the web.

1/31/2013

Keywords

Human computation, STEM innovation and education, epistemology, dialogic games, Karl Popper, mechanism design, social welfare, logic, defense strategies, games and quantifiers, virtual communities.

1. FROM CFP

From call for papers:

Collective intelligence, collaborative production, and social co mputing

Knowledge, education, and scholarship on and through the Web

People-driven Web technologies, including crowd-sourcing, open d ata, and new interfaces

Purpose (of a player playing SCG in an educational setting) : Gain Knowledge about a particular domain. And find out how their total knowledge compares to their peers. Purpose (of a player playing SCG in an R&D setting) : Participate in advancing science. Purpose (of a lab designer in an educational setting) : Encourage students to gain knowledge in a particular domain. Purpose (of a lab designer in an R&D setting) : foster R&D in a particular domain.

2. SEMANTIC GAMES FOR PREDICATE LOGIC

Adapted from SEP

@InCollectionsep-logic-if, author = Tulenheimo, Tero, title = Independence Friendly Logic, booktitle = The Stanford Encyclopedia of Philosophy, editor = Edward N. Zalta, howpublished = `http://plato.stanford.edu/archives/sum2009/entries/logic-if/`, year = 2009, edition = Summer 2009,

In addition to two players P1 and P2, there are two roles: verifier and falsifier. Initially, P1 is falsifier and P2 is verifier. For every predicate logic formula ϕ, model M, and variable assignment g, a two-player zero-sum game $G(\phi, M, g)$ between player P1 and P2 is defined.

$refute(c, P1, P2) = G(\phi, M, claimParameters$, where $c = (\phi, M, claimParameters)$.

1. If $\phi = R(t_1, ..., t_n)$ and $M, g \models R(t_1, ..., t_n)$, the verifier wins; otherwise the falsifier wins.
2. If $\phi = !\psi$, the rest of the game is as in $G(\psi, M, g)$, where P1 has the role that P2 had in $G(\phi, M, g)$, and vice versa.
3. If $\phi = (\psi \wedge \chi)$, the falsifier chooses $\theta in \{\psi, \chi\}$ and the rest of the game is as in $G(\theta, M, g)$.
4. If $\phi = (\psi \vee \chi)$, the verifier chooses $\theta in \{\psi, \chi\}$ and the rest of the game is as in $G(\theta, M, g)$.
5. If $\phi = \forall x \psi$, the falsifier chooses an element a from M, and the rest of the game is as in $G(\psi, M, g[x/a])$.
6. If $\phi = \exists x \psi$, the verifier chooses an element a from M, and the rest of the game is as in $G(\psi, M, g[x/a])$.

What is a model? http://plato.stanford.edu/entries/modeltheory-fo/

K a signature

structure of signature K: ONE domain A

3. THE STRUCTURE OF SCG

We use a standard many-sorted structure to describe the structure of an SCG game domain.

We reuse standard definitions from predicate logic. A signature Ω consists of a non-empty set of sorts $\mathcal{S}$ together with a set of function symbols $\mathcal{F}$ and a set of predicate symbols $\mathcal{P}$. The function and predicate symbols are equipped with arities from $\mathcal{S}^*$ in the usual way. For example, if the arity of $f \in \mathcal{F}$ is $S_1S_2S_3$, then this means that the function f takes tuples consisting of an element of sort S_1 and an element of sort S_2 as input, and produces an element of sort S_3.

To be submitted for review to the ACM Web Science 2013 Conference in Paris, May 2-4. 10 pages limit. Due Feb. 1, 2013

For the semantics of a signature, we have a standard notion of an Ω-$structure$ $\mathcal{A}$, which consists of non-empty and pairwise disjoint domains A_S for every sort S, and $\mathcal{A}$ interprets function symbols f and predicate symbols P by function $f^{\mathcal{A}}$ and predicates $P^{\mathcal{A}}$ according to their arities.

For the SCG-signature Ω we have the sorts Instance, Solution, InstanceSet, Quality (Rationals in $[0, 1]$), Claim, Protocol, Role. The function symbols and their arities are:

```
quality(Instance,Solution,Quality),
getInstanceSet(Claim, InstanceSet),
getProtocol(Claim,Protocol).
getQuality(Claim,Quality).
```

The predicate symbols and their arities are:

```
valid (Instance,Solution),
belongsTo (InstanceSet, Instance),
refuted(Claim,
  List(Quad(Role,Instance,Role,Solution))),
wf(Instance),
wf(InstanceSet),
wf(Solution),
wf(Claim).
```

```
etc.
```

```
wf stands for well-formed.
```

4. REDIRECT PAPER TO CROWDSOURCING

New title: SCG Court: A Crowd Sourcing Platform for Scientific Innovation using Unreliable Scholars

We use game semantics as the foundation of our crowdsourcing approach. Game semantics is used as a voting by justification approach. You can't just vote: the claim is true but you must also successfully defend the claim in the role of verifier.

Origins of GTS (Game Theoretical Semantics):

In the late 1950s Paul Lorenzen was the first to introduce a game semantics for logic. At almost the same time as Lorenzen, Jaakko Hintikka developed a model-theoretical approach known in the literature as GTS. We use Hintikka's approach. cite this

http://plato.stanford.edu/entries/logic-if/

and this:

Hintikka (1968), Language-Games for Quantifiers, in American Philosophical Quarterly Monograph Series 2: Studies in Logical Theory, Oxford: Basil Blackwell, pp. 46-72; reprinted in Hintikka 1973b, Ch. III.

In http://www.logic.rwth-aachen.de/pub/graedel/backandforth.pdf Back and Forth Between Logic and Games

The idea that logical reasoning can be seen as a dialectic game, where a proponent attempts to convince an opponent of the truth of a proposition is very old. Indeed, it can be traced back to the studies of Zeno, Socrates, and Aristotle on logic

In http://plato.stanford.edu/entries/logic-games/ BacForGam

There is also a kind of back-and-forth game that corresponds to our modal semantics above in the same way as Ehrenfeucht-Fraisse games correspond to Hintikka's game semantics for first-order logic.

4.1 Substantiation Game (SG)

A key contribution of our crowdsourcing approach is the substantiation game which allows us to evaluate scholars reliably. We define the SG in extensive form.

We assume a claim c is proposed, either by a scholar or the platform.

RG(c,P1,P2) is a refutation game played based on the formula of the claim. The second position denotes the verifier role and the third the falsifier role. Refutation game r returns the winner: P1 or P2.

When a scholar is blamed it ran into a contradiction of the following two kinds: The scholar s wants to be (1) verifier and s predicts a win, but s lost (VerifierLost) or (2) falsifier and s predicts a win, but s lost (FalsifierLost).

- means no blame.

There are two kinds of substantiation games: either the two scholars conflict in their view on a claim or they agree. If they conflict, a conflict resolution game will be played as defined by the following table (game tree):

- Conflict Resolution Games

P1	P2	ref game	winner	contradiction
v	f	RG(c,P1,P2)	P1	P2
v	f	RG(c,P1,P2)	P2	P1
f	v	RG(c,P2,P1)	P1	P2
f	v	RG(c,P2,P1)	P2	P1

v: wants to play verifier role. f: wants to play falsifier role.

If the two scholars agree, two testing games will be played as defined by the following two tables. If P1 and P2 agree (on being the verifier or falsifier), they play RG(c,P1,P2) and RG(c,P2,P1). If P1 is the verifier, P1 is **under test** in RG(c,P1,P2). If P2 is the falsifier, P2 is **under test** in RG(c,P1,P2).

- Testing Games

P1	P2	ref game	winner	contradiction
v	v	RG(c,P1,P2)	P1	-
v	v	RG(c,P1,P2)	P2	P1
v	v	RG(c,P2,P1)	P1	P2
v	v	RG(c,P2,P1)	P2	-

P1	P2	ref game	winner	contradiction
f	f	RG(c,P1,P2)	P1	P2
f	f	RG(c,P1,P2)	P2	-
f	f	RG(c,P2,P1)	P1	-
f	f	RG(c,P2,P1)	P2	P1

The scholar who wins gets +1 point while the player who loses gets -1 point. The game is zero-sum. When there is agreement and nobody gets blamed, nobody gets a point. We call this the standard payoff function. The only way to make a point is to push the other scholar into a contradiction where the scholar does not substantiate its decision of being verifier or falsifier. The game is very much based on refutation in the spirit of Popper.

SCG has the following broad and diverse applications: 1. Crowdsourcing in Formal Science 2. Teaching Formal Science 3. Software Development Process for Computational Problems 4. (Potential) Automation Gaps in Theorem Provers An important contribution is to step from the traditional refutation games of classical and independence-friendly logic to binary games which extend the refutation games with claim assertions to detect contradictory behavior of the scholars. This allows us to rank the scholars reliably and gives us better evidence about which claims are true. The contradictory behavior that we can detect, is based on clever mechanism design. The essence of SCG is to make assertions about claims:

either true or false. But each assertion needs to be substantiated constructively. If it is not substantiated, the scholar is blamed for reaching a contradiction. The game is designed in such a way that a super perfect player can always avoid running into a contradiction. A super perfect player is a player that makes always correct decisions and who can substantiate all its decisions.

A strategy for a clever scholar might be to always make the same assertion as the opponent. And then to go easy with the substantiation by tacit agreement with the opponent. Nobody will be blamed but also nobody will win a point with this strategy. You can only win a point by driving the opponent into a contradiction.

Is it easy to come up with the substantiating constructions? Yes, we illustrate 4 interesting binary game designs below and many more are possible.

5. UNRELIABLE SCHOLARS

Managing unreliable workers is a standard problem in crowdsourcing systems. Many solutions have been proposed [16] and significant worker resources are used to detect and control unreliable workers. However, SCG has special properties which we can exploit to come up with a better solution. A clever mechanism design based on SG allows us to reliably evaluate workers.

A standard approach in crowdsourcing is to use a gold standard for labeling (if it is available) and to measure the performance of a worker on the gold standard (objective ground truth) [13, **?**, **?**]. In SCG it is best to reverse the standard approach: We have a direct way to measure the quality of a worker. We use this quality to assign labels to claims (true or false or optimum). We assume in this section that the claims are independent. If there are implication relationships between claims we have additional information regarding the truth of the claims.

In SCG we can reliably measure the number of contradictions produced by a scholar. The more contradictions, the weaker the scholar. The art is to define the concept of contradiction properly. These are two attempts which don't work: a scholar is contradictory (1) if it proposes a false claim. (2) if it disputes a true claim. The reason is that we don't know whether the claim is true or false.

The right way to define the concept of **contradictory scholar** is as follows: A scholar s is **contradictory** if one of the two contradictory states are reached during a binary game: VerifierLost or FalsifierLost.

These two states are exactly the ones where scholar s gets blamed in SCG. Using the standard payoff function, these are the states where there is a negative payoff for s.

5.1 Risk of Blame for Unreliability

How is this changed now in the new game. What is now conditional blame?

A scholar s is **unreliable**, if s performs one of the following actions: New: If s is the verifier of a false claim, s risks VerifierLost. If s is the falsifier of s true claim, s risks FalsifierLost.

This is the primary mechanism used by SCG to deal with unreliable scholars. The secondary mechanism is that most game outcomes lead to blame and the outcomes which do not lead to blame require successful support of the claim or successful refutation of the claim.

Because an unreliable scholar risks to be blamed, SCG discourages wrong information.

Fact: unreliable scholars risk to be blamed.

of course: contradictory scholars are blamed.

In summary, decisions/actions that contribute wrong information are potentially blamed, and decisions/actions that are contradictory are blamed.

5.2 Avoiding Contradictions

It is important that those two contradictory states FalsifierLost, VerifierLost can be avoided in principle by a perfect scholar. A simple case analysis shows that this property holds:

- FalsifierLost: If c is true, be the Verifier and defend c. If c is false, refute the claim.
- VerifierLost: If c is true, defend the claim. If c is false, be the Falsifier and refute c.

So we know that we can never be forced into a contradiction if we play well.

SCG is a blame avoiding game. You make all your decisions according to the game tree in such a way that you don't contradict yourself and therefore you will not be blamed. If you are blamed, your reputation goes down and you will have less influence in deciding the truth of claims. If you are not blamed, your reputation goes up or stays the same and you will have more influence in deciding the truth of claims.

reputation -> reliability

During binary games we accumulate evidence about the reputation of the scholars and the truth of the claims. We do this in the presence of unreliable scholars.

5.3 Perfect Scholars

A scholar s is **perfect**, if it never shows contradictory behavior.

A perfect scholar can be used to decide whether a claim is true or false. Put the perfect scholar into the Opponent role and Proponent proposes a claim c. The Opponent will decide a or d correctly because otherwise it could be forced to show contradictory behavior. The perfect scholar can be used to defend a true claim it agreed with and to refute a claim it disputed.

$!s$ is the "other" scholar. If s is P then $!s$ is O and if s is O then $!s$ is P.

5.4 Rating Systems for Scholars

Our plan for collecting evidence about a claim c is as follows. We assign to every scholar S a reliability r(S) in [0,1]. Then we let S vote with justification in its game behavior for true or false. We assign a weight r(S) to its vote.

We now get into the domain of rating systems for games with wins, losses and draws. A good survey is given in [5]. rating measure actual skill of scholars grade of scholar This is a controversial subject and there are many algorithms that can be used unweighted weighted means using the last n results

measure of performance measure of ability

5.4.1 From Contradictions to Game Actions

With the reliability $rel(s)$ of scholar s we measure how reliable the scholar is to avoid contradictions. To avoid contradictions, a scholar must be good at the following subtasks: (1) proposing true claims, (2) correctly judging claims (agree or dispute) and (3) reliably refuting claims it disputed and (4) reliably defending claims it agreed with. We cannot easily measure how well the scholar does on the above subtasks but only on the overall task to avoid contradictions. But we know that to avoid contradictions, the scholar must do well on *all* the above subtasks. Therefore, if a scholar is reliable on the overall task it must be reliable on the subtasks as well.

5.4.2 Limitations of Rating Systems

[5] is very critical of the value of player rating systems:

> A grade is merely a general measure of a player's performance relative to that of certain other players over a particular period. It is not an absolute measure of anything at all.

In SCG we are interested in correctly classifying the claims into true and false claims and in the “clever constructions” that are needed to defend the true claims and to refute the false claims. The “clever constructions” are most likely owned by the reliable scholars who know how to avoid contradictions.

5.4.3 Two Proposals for Player Rating Computation

We present two ways to compute a player rating.

- Order Dependent Rating.

  The first approach is sequential and depends on the order of the binary games. We first compute a reputation for each player which is then turned into a strength which is normalized between 0 and 1.

  Initially, all scholars s have the same initial reputation $rep(s) = 1$. The reputation of a scholar s is updated as follows: For all three possible contradictions,

  $$rep(!S) = rep(!S) + rep(S).$$

  The higher the reputation of a scholar the better it is to avoid contradictory behavior.

  We generalize the reputation computation to take positive payoff functions into account: We assume the payoff function $payoff(s)$ to be non-negative. We update the reputation of P and O after a binary game as follows:

  $$rep(O) = rep(O) + payoff(O) * rep(P)$$

  $$rep(P) = rep(P) + payoff(P) * rep(O)$$

  Consider the simple payoff function $Spayoff(s)$ defined by: when s not blamed, then $Spayoff(s) = 1$ and when s is blamed, then $Spayoff(s) = 0$. If we use $Spayoff$, then $rep(s)$ is of the simple form introduced above: if s is not blamed then $rep(s) = rep(s) + rep(!s)$. $rep(s)$ stays the same when s is blamed.

  We define the likelihood that a scholar s is strong: $strength(s)$. We consider all scholars and compute their maximum reputation rmax. The likelihood that scholar s is strong is given by $strength(s) = rep(s)/rmax$. Note that $strength(s) = 1$ does not imply that s is perfect because it is possible that s got into a contradiction and has not accumulated the maximum possible reputation. $strength(s) = 1$ means that s is among the best in the given set of scholars.

- Order Independent Rating.

  Ahmed's fixpoint computation for reliability. $rel(s) =$

We prefer the second approach and use $rel(s)$ for the rest of the paper.

5.4.4 Alternative Scholar Rating

Count $Bad(s)$ how many s-NDA, s-NDP and s-NRD outcomes (contradictions) scholar s achieved. Count $Good(s)$ how many no blame outcomes s achieved where s successfully supported a claim. $Good(s)/(Good(s) + Bad(s))$ is the reliability of scholar s. Note that a perfect scholar s can force $Bad(s) = 0$.

5.5 Voting with Justification

How should we accumulate evidence about the truth of a claim? We use a voting with justification approach. You cannot just vote "yes, this claim is true" but you must support your vote with your actions in the refutation game.

Voting with justification All voting by a scholar is influenced by the scholar's reputation. The higher the reputation, the more weight has the vote.

We have two scholars $s = P$ or $s = O$.

We use the terminology from Figure 3 of [22]:

- **asO** agreement followed by support by O (no blame)
- **arP** agreement followed by refutation by P (O-NDA)
- **drO** dispute followed by refutation by O (P-NDP)
- **dsP** dispute followed by support by P (O-NRD)

stronger means: higher reputation

5.5.1 Simple Voting for Claims

We accumulate information about whether a claim is true or false. Each claim c has a positive real number c_T and c_F associated with it. The higher c_T the more likely c is true. The higher c_F the more likely c is false. If $c_T > c_F$ then

$$L(c) = (c_T - c_F)/c_T$$

is the likelihood that c is true. And $1 - L(c)$ the likelihood that c is false. If $c_F \geq c_T$ then

$$L(c) = (c_F - c_T)/c_F$$

is the likelihood that c is false. and $1 - L(c)$ the likelihood that c is true.

- The claim is supported. Scholar s votes claim c to be true because s defended it against $!s$. Support happens for **asO** and **dsP** and we update c_T as follows:

  $$c_T = c_T + rel(O) + rel(P).$$

  The stronger P or O the more likely c is true.

- The claim is refuted. Scholar s votes claim c to be false because s refuted it. Refutation happens for **arP** and **drO** and we update c_F as follows:

  $$c_F = c_F + rel(O) + rel(P).$$

  The stronger P or O the more likely c is false.

5.5.2 Better 3-Component Voting for Claims

There is a better way of voting for the truth of claims which takes more context into account than just the fact whether a claim was refuted or defended.

The 3-Component voting is based on the 3 levels of the binary game:

1. The proponent P votes the claim to be true. Weight of vote: $rel(P)$.
2. If O decides **a** (agree), it votes for the claim to be true. If O decides **d** (dispute), it votes for the claim to be false. Weight of vote: $rel(O)$.
3. The game makes a vote based on the outcome as follows (we used only this above): If ***s**s (support by s), the game votes the claim to be true. if ***r**s (refute by s), the game votes claim to be false. Weight of vote: $rep(!s)$.

For the four cases we get:

- **asO:** $c_T = c_T + rel(P) + rel(O) + rel(P)$, c_F unchanged
- **arP:** $c_T = c_T + rel(P) + rel(O), c_F = c_F + rel(O)$
- **drO:** $c_T = c_T + rel(P), c_F = c_F + rel(O) + rel(P)$
- **dsP:** $c_T = c_T + rel(P) + rel(O), c_F = c_F + rel(O)$

compare to: http://www.ccs.neu.edu/home/lieber/papers/SCG-Paper/main.pdf

Questions:

1. Is the above an application of Dempster-Shafer theory:

Combination of Evidence in Dempster-Shafer Theory by Kari Sentz: http://www.sandia.gov/epistemic/Reports/SAND2002-0835.pdf We have epistemic uncertainty because our uncertainty results from the lack of knowledge.

2. Does Dempster-Shafer Theory suggest a better solution for computing the likelihood that a claim is true?

```
http://www.umiacs.umd.edu/labs/cvl/pirl/vikas/publications/raykar_JMLR_2010_crowds.pdf
Learning from Crowds

Paper should be ready one week
before due date: Jan. 25, 2013
and will be sent to Magy on
that date or sooner.

Ahmed's task (Jan. 8):
Write section:
Mechanism Design for Crowdsourcing
(Formerly: The SCG Design Pattern
with Applications)
Four subsections:
1. Optimization Labs
2. Agreement with two refutations
3. Perfect Labs
4. Less Competition (to support brainstorming)
(formerly less competitive payoff ...)

Ahmed proposes a good way to deal with implied games.
(games where some of the decisions
have been made)

Ahmed proposes a good way to deal
with Lab decompositions.
What is the difference between lab decompositions
and problem decompositions.
In lab decompositions we don't DIVIDE
AND CONQUER? We TRANSFORM AND CONQUER?

Karl works out three examples of lab decompositions.
```

A very important property of our approach to crowdsourcing is that we take good care of the crowdsourcing workers.

- feedback: when points get deducted there is a demonstrated reason.
- examples: see knowledge base of claims and history of claims
- get rewarded for breakthroughs

Crowd Sourcing to Distinguish Good from Bad

The Scientific Community Game as A Crowdsourcing Platform to Distinguish Good from Bad

Domain of requests (instances) and responses (solutions). Responses are checked against valid(request,response). Claims are about the relationship between requests and responses. Claims are divided into good and bad claims. Good claims are claims that are predominantly defended. Bad claims are claims that are predominantly refuted. Refutation is the complement of defense and is based on the requests and responses exchanged.

Want to build artifact: good claims. And the corresponding techniques to defend them.

Definition from Communications of the ACM: A CS system enlists a crowd of users to explicitly collaborate to build a long lasting artifact that is beneficial to the community.

Also CACM: enlists a crowd of humans to help solve a problem defined by the system owners.

The White Paper Version: Crowdsourcing is the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call.

Four challenges:

(How to recruit and Retain Users?)

What contributions can users make? provide requests and responses, propose and oppose claims about requests and responses

How to combine user contributions to solve the target problem.

Describe in detail how this is done in SCG: bottom-up or top-down. lab decompositions and meta claims (subject to refutations). strengthening, correcting mistakes.

Break down a lab into simpler labs. Extend basic game:

How to evaluate users and their contributions. breakthroughs learning activity how many contradictions

Describe in detail how this is done in SCG

Related Work:

Group organization. Warren Bennis in his book: Organizing Genius, The Secrets of Creative Collaboration, says: "you create an atmosphere of stress, creative stress, everyone competing to solve one problem."

Crowd IQ: measuring the intelligence of crowd sourcing platforms Web Science 2012

we have our own way of measuring crowdIQ: count contradictions.

The CrowdLang paper by Abraham Bernstein at Web Science 2012

CrowdForge: Crowdsourcing complex work (2011) ACM from CMU http://ra.adm.cs.cmu.edu/anon/usr/ftp/anon/hcii/CMU-HCII-11-100.pdf

We identify the coordination requirements necessary to crowdsource complex tasks, and describe a framework to support a variety of task types. The framework systematically breaks complex problems down into simpler tasks by creating subtasks that define and create other subtasks and distributes these tasks to workers. Output from subtasks can be evaluated and consolidated via additional outsourced tasks.

6. FUZZINESS

http://jcr.sagepub.com/content/50/1/28.full.pdf+html
contains interesting references

7. NEGATION

It is important to have negation for claims and a refutation-based treatment of negation. For example, the set of claims of a lab should be closed under negation using the standard negation operator (!). We need to define what it means to refute !c in terms of what it means to refute c.

So far we have not explicitly defined negation, but implicitly it is used in the SCG. Consider the decision to agree with a claim. In this case the proponent (P) and opponent (O) play the refutation game refute(c,O,P). This has intuitively the intention that O cannot bluff and it must defend the claim against P to substantiate the agreement. In other words, we could say that O must refute the negation of c: refute(!c,P,O).

The refute function has two argument positions for the players. The second position is for the verifier role and the third one for the falsifier role. The player in the verifier role is trying to make the refutation predicate true while the player in the falsifier role is trying to make it false.

The following two situations are equivalent: refute(c,O,P), O wins (O "verifies" c to be true), and refute(!c,P,O), O wins (O "falsifies" !c to be false).

Therefore the rule for negation is a role switch, similar to the role switch as originally proposed by Hintikka in 1968 for Independence Friendly Logic [34].

In the future we talk about negated claims. A useful lab is a bivalent lab with two claims: c and !c and the purpose of the lab is to determine which of the two claims is true.

8. STANDARD BINARY GAMES

To simplify the work of a lab designer, We have standard refutation protocols to choose from. For the same reason, we offer standard binary games.

I am not sure about the name: binary game. It is an important part of a binary game.

The substantiation part says what needs to be done constructively to support the assertion.

8.1 Binary Game 1

P is given claim c.

- assertion T

  P asserts assertTrue(c).

  Substantiation: P defends c against O, i.e., P wins refute(c,P,O).

- assertion F

  P asserts: assertFalse(c)

  Substantiation: P refutes c against O, i.e., P wins refute(c,O,P).

8.2 Binary Game 2

P is given a claim c.

- assertion Lopt

  P asserts: c is locally optimum.

  Substantiation: P defends c against O, i.e., it wins refute(c,P,O). P proposes a stronger c, called c', and refutes c' against O, i.e., P wins refute(c',O,P).

- assertion !Lopt

  P asserts: c is true but not locally optimum.

  Substantiation: P defends c against O, i.e., it wins refute(c,P,O). P proposes a stronger c, called c', and defends c' against O, i.e., P wins refute(c',P,O).

- assertion T

- assertion F

8.3 Binary Game 3

P is given a claim c.

- assertion indet

  P asserts that claim c is indeterminate.

  Substantiation: If refute(c,O,P) is played n times, the probability that O wins n times is 2^{-n}.

- assertion T (see above)

- assertion F (see above)

8.4 Binary Game 4

Consider two labs Lab1, Lab2 and the mapping T from Lab1 to Lab2. Instances, solutions, claims are all mapped by T. Consider the image claim T(c) in Lab2 for claim c in Lab1.

- assertion 1

  P asserts that if claim T(c) is true in Lab2 then c is true in Lab1.

  Substantiation: If P is given a defense of T(c) in Lab2, P can construct a corresponding (under T) defense history of c in Lab1.

- assertion 2

  P asserts that if claim T(c) is false in Lab2 then c is false in Lab1.

  Substantiation: If I am given a refutation of T(c) in Lab2, I can construct a corresponding (under T) refutation history of c in Lab1.

9. OUR THESIS

Gamification of innovation in formal sciences has a formal foundation with useful applications.

Contributions of this paper:

Formulation of SCG Design Patternand illustration with 3 examples: optimization, different agreement, master scholar.

Concept of blame strength and how it translates to payoff.

Concept of lab reductions and how they contribute to problem solving.

Convergence to optimum claim. What are the necessary preconditions?

intrinsically motivating instruction by Tom Malone http://mailer.fsu.edu/ jkeller/I

10. POSITIVE TERMINOLOGY APPROACH TO SCG

Terminology change: opponent -> partner.

Create learning opportunities for partner by creating outcomes where the partner is contradictory. Contradictory bevavior is a seed for learning. conditionally or absolutely.

Goal of game: create learning opportunities?

11. NEGATION

We introduce a negation operator for labs that maps claims to claims union negated(claims). Given a claim c, the negation of c is the claim where ... switch P and O negate refutation predicate. A lab is closed under negation if for every claim c, the claim not(c) is also in the lab.

Protocol with refutation predicate (P,O,predicate) (O,P,!predicate)

The simple rule for claim negation is: the domain stays the same and, in the protocol, the roles of Alice and Bob are reversed and a defense is changed into a refutation.

In the following we assume that all labs are closed under negation.

refute(c,P,O)) = defend(!c,O,P) Defending a claim c has the same difficulty as refuting its complement !c.

negation is needed agree(c) dispute(c) agree(!c) strengthen(c,c')

agree(!c) = dispute(c) ??

Negation in IF logic:

Hintikka proposed to interpret negation in terms of role shift. When semantic games are formulated for arbitrary first-order formulas, in addition to the two players there are two roles to be considered: ÂŚverifierÂŠ and ÂŚfalsifier.ÂŠ In the beginning, player 1 occupies the role of ÂŚfalsifierÂŠ and player 2 that of ÂŚverifier.ÂŠ Negation is then interpreted by transposing the roles: the player whose current role is ÂŚverifierÂŠ assumes the role of ÂŚfalsifier,ÂŠ and vice versa.

Negation. By clause (1) of the syntax, the negation sign may only appear as prefixed to an atomic formula. Conceptually there is no reason for this restriction; clause (1) may be replaced by a clause laying it down that any formula f of FO[t] is a formula of IF first-order logic; while the rest of the clauses are kept intact (cf. Sandu 1994, 1996). Let us refer to the syntax with arbitrary occurrences of the negation sign as liberalized syntax.

It remains to be told how such negation signs are interpreted in GTS. Two roles must be added as a new ingredient in the specification of the games: those of ÂŚverifierÂŠ and ÂŚfalsifier.ÂŠ Initially player 1 has the role of ÂŚfalsifierÂŠ and player 2 that of ÂŚverifier.ÂŠ The roles may get switched, but only for one reason: when negation is encountered. Due to the introduction of the roles, all clauses defining semantic games must be rephrased in terms of roles instead of players. More specifically, it is the player whose role is ÂŚverifierÂŠ who makes a move for tokens of (?/?y1,ÂĚ,?yn) and (?x/?y1,ÂĚ,?yn), and similarly the player whose role is ÂŚfalsifierÂŠ who moves for the tokens of (?/?y1,ÂĚ,?yn) and (?x/?y1,ÂĚ,?yn). When a formula ? is encountered, the players change roles and the game continues with ?. Finally, if the encountered atomic formula is true, ÂŚverifierÂŠ wins and ÂŚfalsifierÂŠ loses, otherwise the payoffs are reversed. For a more detailed description, see the supplementary document. The negation works as one would expect: the sentence f is false in M iff its negation f is true therein. Similarly, a sentence f is true in a model M iff its negation f is false in M (cf. Sandu 1993).

The negation is variably referred to as strong negation, dual negation, or game-theoretical negation.[33] Due to the failure of bivalence, the logical law of excluded middle fails as well: if f is a sentence non-determined in a model M, then M ? (f ? f). In what follows, the original syntax and semantics given in Subsections 3.1 and 3.2 will be applied, unless otherwise stated.

12. THE SCG Design Pattern WITH APPLICATIONS

In paper [22] we use the following game design pattern called SCG Design Pattern.

Using Game Goal and Blame Assignment to systematically Design the Payoff Function

In our game design problems (1) there is a game design goal that defines what the game should achieve (2) there are blamable moves that are considered non-productive with respect to the design goal.

The design goal is then translated into a blame assignment that matches the design goal.

Finally, the blame assignment is translated into a payoff function that is fair, sound and competitive with respect to the blame assignment.

Fairness means that if S is not blamed, payoff(S)>=payoff(!S). (!S is the other scholar.)

Soundness means that if S is blamed then payoff(S)<payoff(!S). Or soundness means that if S is blamed then there is a chance that S will have a negative payoff. (There are different variants of soundness depending on the different kinds of blamable moves.)

Competitiveness means that the payoff is higher for the winner.

The first important goal in SCG is not to contradict yourself. You contradict yourself, if you make a decision which has an implied assumption but then you don't satisfy that assumption. For example, if you decide to dispute a claim, the implied assumption is that you will refute it successfully. If you don't, you contradict yourself. See Figure 7 for all 5 ways to contradict yourself in SCG with Optimization.

A second important goal is not to propose false claims. You risk being caught when you do.

12.1 Application to Optimization Labs

Useful for learning: want to converge to optimum claim.

12.1.1 Blame and Payoff Table

10 explains the issues described in the following paragraph:

Regarding blame, blame is a technical term that we carried over from PL. The point is that we want to formally verify that our game design isn't futile especially that a part of the final game tree is provided by lab designers. Our approach consists of labeling certain edges / nodes of the game tree with additional properties (for example that an edge represents a blamable action or that we cannot distinguish between actions taken at a particular node). Then assert certain generic properties (that involve those newly added labels) about the the overall tree. These generic formal properties define what makes the game interesting (or non futile). We couldn't find such generic definitions of game interestingness in the literature. And would like to get your input on that. (from Ahmed's email)

From where come the blamable actions? They are implied by the decisions made. Each decision has an expected result: When you are accepted by a PhD program, people expect that you get your PhD. When you agree with a claim, you are expected to defend it successfully. When you propose a claim, you are expected to defend it successfully or to refute a stronger claim. When you dispute a claim, you are expected to refute it. When you strengthen a claim, you are expected to defend the strengthened claim.

If you fail to meet the expectation, you are called contradictory and you get blamed.

If you fail to meet the expectation, you get blamed. Your goal in the game is to teach your opponent by bringing him or her into a situation where it gets blamed absolutely (column oB) or conditionally (columns fB and nB).

In Figure 10, in columns fB and nB we give a row number which indicates how to translate conditional blame into a positive payoff for the opposer. The table has 3*6+1=19 rows. If a false claim is proposed, the best action is to dispute it and to successfully refute it (row 8: the header row is row 1). If a non-optimal claim is proposed, the best action is to strengthen it and to defend the strengthened claim (row 15).

add a new decision possibility for dec: s = strengthen

update blame justification

size of table: Consider the table 7 which describes the generalization for optimization.

This table seems very useful as we see all the information in one table not spread out between a game tree and a table.

A claim is either true or false. A true claim should be optimum.

Figure 11 describes all learning opportunities. There are two levels of learning opportunities: level 1 in column 1B and level 2 in column 2B. Blame is not only assigned for claim choices (proposing a false or non-optimal claim, level 1) but also for decisions (e.g., disputing an optimal claim, level 2).

1B: row number that blames choice by forcing loss

2B: row number that blames decision by showing better decision that avoids loss. We show the line number for the case where there is an improvement if a better decision is made.

Column 2B:

Ta is blamed because TssO (row 15)is guaranteeing a win for O.

Td is blamed because TssO is guaranteeing a win for O.

T-optd is blamed because T-OptasO is avoiding a loss for O.

T-opts is blamed because T-OptasO is avoiding a loss for O.

Ahmed talks about CTL expressible properties that tree must have. Can they be expressed with such row numbers?

What are the constraints that must hold? They are in Figure 8.

12.2 Application to Agreement with two Refutation Games

Has first the flavor of a regular dispute.

12.2.1 Blame and Payoff Table

add a second out2 column used for agreement only.

see Figure 9. The agreement protocol consists of two applications of the refutation protocol with the provision that all solutions are only revealed at the end of the protocol.

advantages

Two applications of the refutation protocol with reversed roles leads to more testing of claims and scholars. The game is more balanced: P and O are blamed in the oB* columns on two outcomes while before only O could be blamed in the oB column.

disadvantages

The cost is higher.

Casper: no negative payoff. Exception both get a negative payoff:

```
0   0
1  -1
-1  1
-1 -1

becomes
0   0
1   0
0   1
a   a
where a = -1/4.
```

12.3 Application to Perfect Labs

We call a lab perfect if the lab designers know which claims are true and which are false. In this case the blame can be targeted more directly because there is no uncertainty about whether a claim is true or false. This applies often during learning where the lab designer (teacher) has more knowledge than the students.

See Figure 12. Is the payoff fair and sound? Complete the table. There is a need to have a weight on the blame (strength of the learning opportunity).

12.3.1 Blame and Payoff Table

13. LESS COMPETITIVE PAYOFF FOR LEARNING

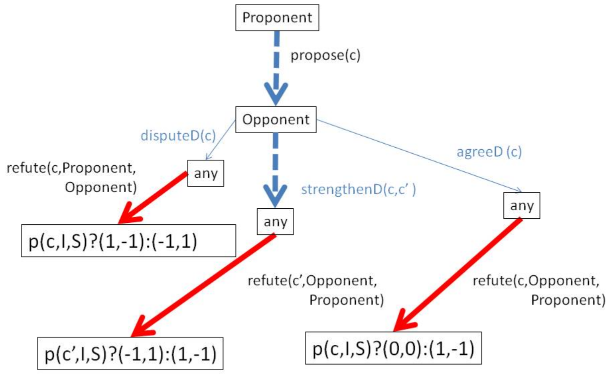


Figure 1: SCG Binary Game Tree.

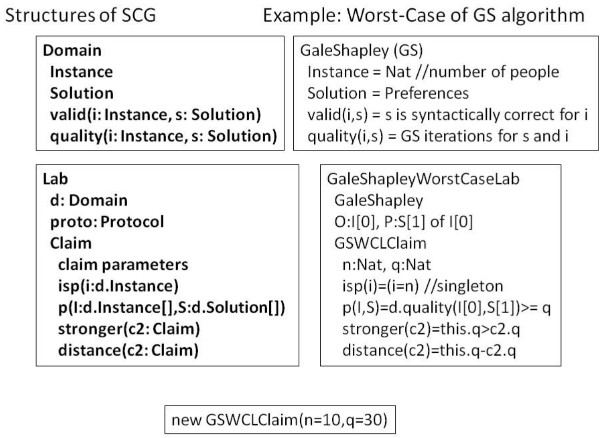


Figure 2: SCG Structure.

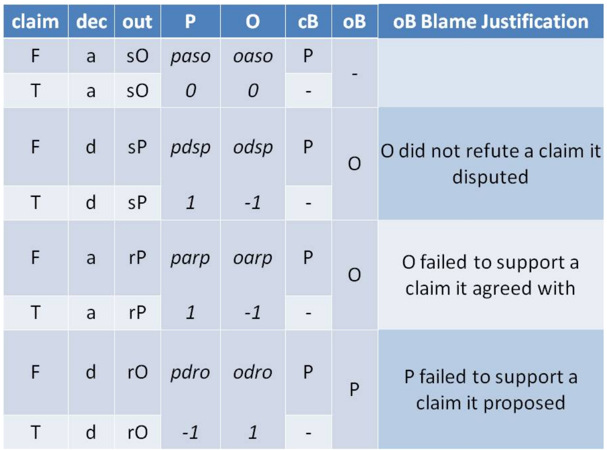
Is this the correct translation from paper [**?**].

refute()? (pdsp,odsp):(parp,oarp) agree()? (paso,oaso):(pdro,odro)

Depending on the application, many meaningful payoff functions can be defined. For example, if SCG is used for creating student interaction in a MOOC, I recommend the following low competition payoff function (see Figure 4. The values after / are for learning. The competitive payoff function is shown before /):

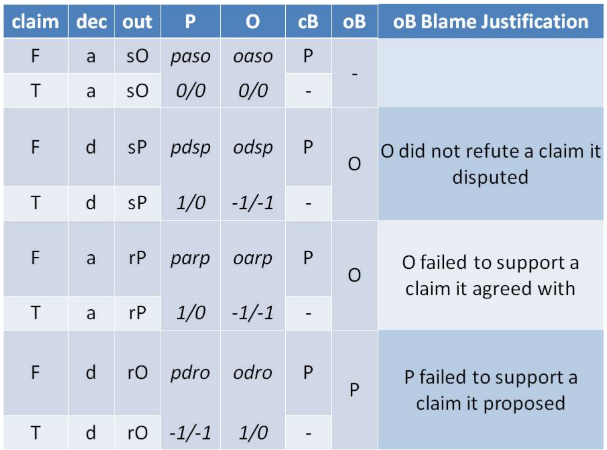
1. refute: $p(c, ...)?(0,0)\ :\ (0,1)$. If the predicate is true, nobody gets a point because we want the Opponent to learn from the Proponent through the refutation protocol. If the predicate is false, the Opponent has won and gets a point.

2. strengthen c to c': $p(c', ...)?(0,1) : (0,0)$. If the predicate is true, the Opponent gets rewarded with one point because he successfully defended the stronger claim. If the predicate is



claim	dec	out	P	O	cB	oB	oB Blame Justification
F	a	sO	*paso*	*oaso*	P	-	
T	a	sO	*0*	*0*	-		
F	d	sP	*pdsp*	*odsp*	P	O	O did not refute a claim it disputed
T	d	sP	*1*	*-1*	-		
F	a	rP	*parp*	*oarp*	P	O	O failed to support a claim it agreed with
T	a	rP	*1*	*-1*	-		
F	d	rO	*pdro*	*odro*	P	P	P failed to support a claim it proposed
T	d	rO	*-1*	*1*	-		

Figure 3: Blame and Payoff Table



claim	dec	out	P	O	cB	oB	oB Blame Justification
F	a	sO	*paso*	*oaso*	P	-	
T	a	sO	*0/0*	*0/0*	-		
F	d	sP	*pdsp*	*odsp*	P	O	O did not refute a claim it disputed
T	d	sP	*1/0*	*-1/-1*	-		
F	a	rP	*parp*	*oarp*	P	O	O failed to support a claim it agreed with
T	a	rP	*1/0*	*-1/-1*	-		
F	d	rO	*pdro*	*odro*	P	P	P failed to support a claim it proposed
T	d	rO	*-1/-1*	*1/0*	-		

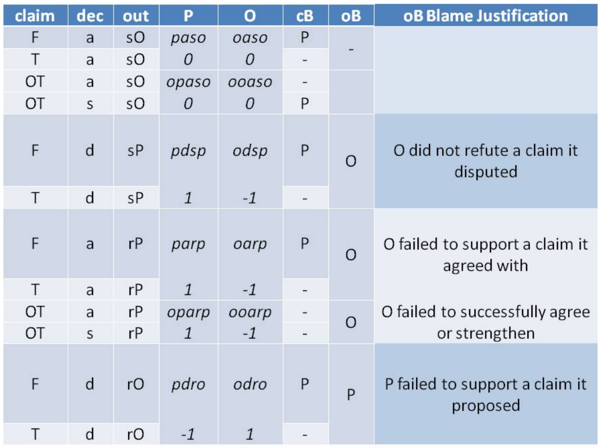
Figure 4: Blame and Payoff Table (Learning)

false, the Proponent has won but does not get a point because we want the Opponent to have cheap opportunities to attack and learn.

3. agree: $p(c,...)?(0,1) : (0,0)$. If the predicate is true, the Opponent has successfully defended the claim and nobody gets a point. If the predicate is false, the Opponent has failed to defend the claim but has gained information to learn. The Proponent earns a point.

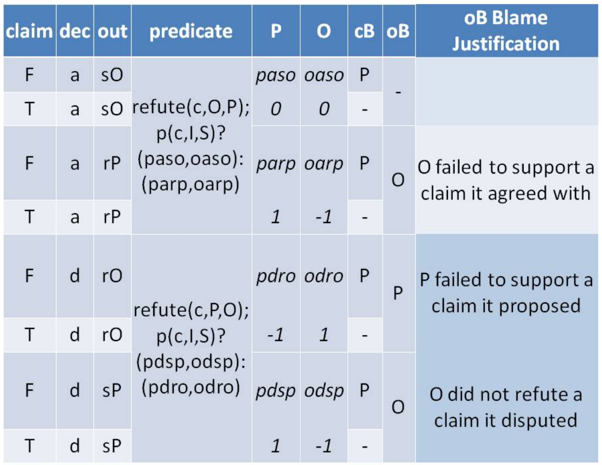
The motivation is that it should be penalty free for students to learn from other students. Not succeeding in refuting a claim is free to the Opponent, while successfully refuting gives a point to the Opponent. Failing to strengthen a claim is free to the Opponent, while successfully strengthening gives a point to the Opponent. Failing to agree with a claim is free to the Opponent, while successfully agreeing gives a point to the Opponent This low competition payoff function has the flavor of a soccer game where only the goals count.

The competitive payoff and the low competition payoff are two examples of payoff functions that promote good behavior in the lab. Other payoff functions are possible.



claim	dec	out	P	O	cB	oB	oB Blame Justification
F	a	sO	*paso*	*oaso*	P	-	
T	a	sO	*0*	*0*	-		
OT	a	sO	*opaso*	*ooaso*	-		
OT	s	sO	*0*	*0*	P		
F	d	sP	*pdsp*	*odsp*	P	O	O did not refute a claim it disputed
T	d	sP	*1*	*-1*	-		
F	a	rP	*parp*	*oarp*	P	O	O failed to support a claim it agreed with
T	a	rP	*1*	*-1*	-		
OT	a	rP	*oparp*	*ooarp*	-	O	O failed to successfully agree or strengthen
OT	s	rP	*1*	*-1*	-		
F	d	rO	*pdro*	*odro*	P	P	P failed to support a claim it proposed
T	d	rO	*-1*	*1*	-		

Figure 5: Blame and Payoff Table (Optimization)



claim	dec	out	predicate	P	O	cB	oB	oB Blame Justification
F	a	sO	refute(c,O,P); p(c,I,S)? (paso,oaso): (parp,oarp)	*paso*	*oaso*	P	-	
T	a	sO		*0*	*0*	-		
F	a	rP		*parp*	*oarp*	P	O	O failed to support a claim it agreed with
T	a	rP		*1*	*-1*	-		
F	d	rO	refute(c,P,O); p(c,I,S)? (pdsp,odsp): (pdro,odro)	*pdro*	*odro*	P	P	P failed to support a claim it proposed
T	d	rO		*-1*	*1*	-		
F	d	sP		*pdsp*	*odsp*	P	O	O did not refute a claim it disputed
T	d	sP		*1*	*-1*	-		

Figure 6: Blame and Payoff Table (with Game Tree)

Instances are only available when they are needed. For example, in the spirit of the Renaissance mathematical competitions between Tartaglia and Fior, if the protocol asks that the Proponent and Opponent deliver each 10 instances, followed by the solution activity. The instances are secret until they are solved.

14. GAME HISTORY

Use it to measure learning.

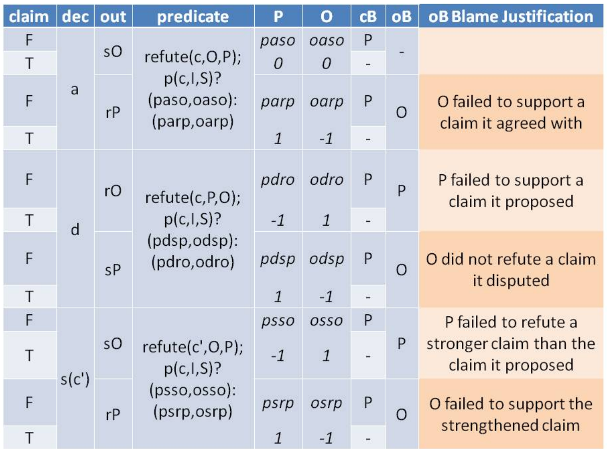
Discuss learning from previous games. In game G1 you disputed claim c and successfully refuted it. In a future game G2 should you dispute it again?

Not necessarily, because you could fail to refute the claim c in G2.

Reasons could be:

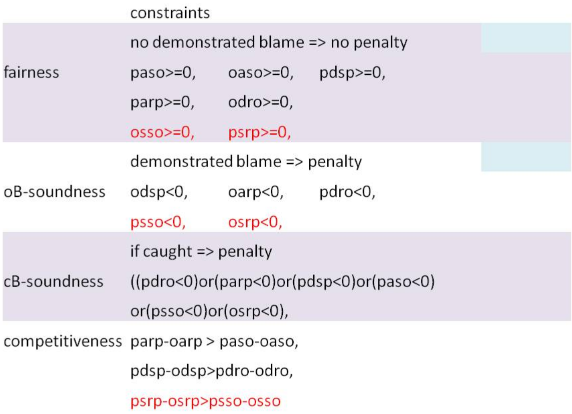
1. The proponent has improved and found a defense strategy for its claim c.

2. Although the claim is false, you lack a systematic refutation



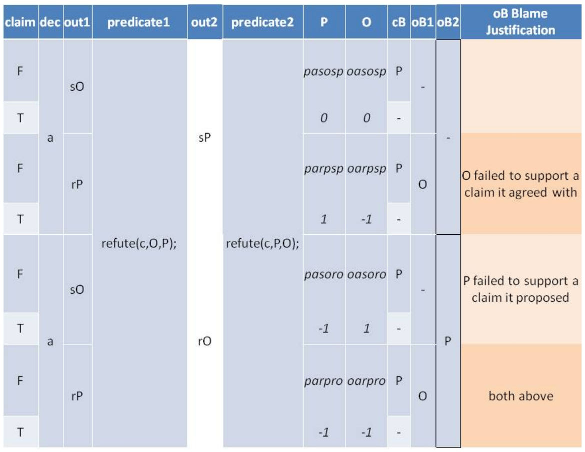
claim	dec	out	predicate	P	O	cB	oB	oB Blame Justification
F	a	sO	refute(c,O,P); p(c,I,S)? (paso,oaso): (parp,oarp)	*paso*	*oaso*	P	-	
T				*0*	*0*	-		
F		rP		*parp*	*oarp*	P	O	O failed to support a claim it agreed with
T				*1*	*-1*	-		
F	d	rO	refute(c,P,O); p(c,I,S)? (pdsp,odsp): (pdro,odro)	*pdro*	*odro*	P	P	P failed to support a claim it proposed
T				*-1*	*1*	-		
F		sP		*pdsp*	*odsp*	P	O	O did not refute a claim it disputed
T				*1*	*-1*	-		
F	s(c')	sO	refute(c',O,P); p(c,I,S)? (psso,osso): (psrp,osrp)	*psso*	*osso*	P	P	P failed to refute a stronger claim than the claim it proposed
T				*-1*	*1*	-		
F		rP		*psrp*	*osrp*	P	O	O failed to support the strengthened claim
T				*1*	*-1*	-		

Figure 7: Blame and Payoff Table (with Optimization)



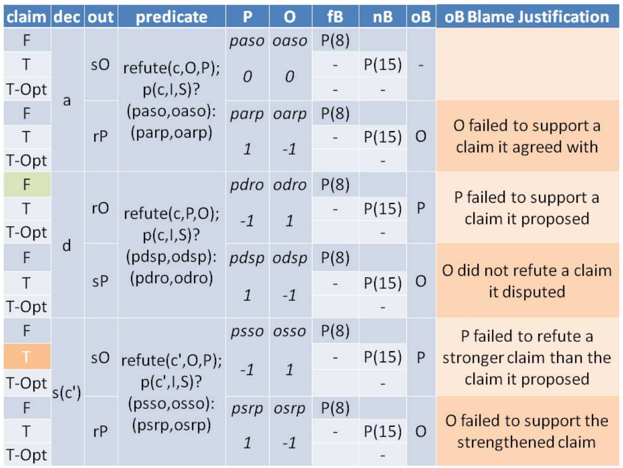
	constraints
	no demonstrated blame => no penalty
fairness	paso>=0, oaso>=0, pdsp>=0,
	parp>=0, odro>=0,
	osso>=0, psrp>=0,
	demonstrated blame => penalty
oB-soundness	odsp<0, oarp<0, pdro<0,
	psso<0, osrp<0,
	if caught => penalty
cB-soundness	((pdro<0)or(parp<0)or(pdsp<0)or(paso<0)
	or(psso<0)or(osrp<0),
competitiveness	parp-oarp > paso-oaso,
	pdsp-odsp>pdro-odro,
	psrp-osrp>psso-osso

Figure 8: Constraints Payoff Design (with Optimization)



claim	dec	out1	predicate1	out2	predicate2	P	O	cB	oB1	oB2	oB Blame Justification
F	a	sO	refute(c,O,P);	sP	refute(c,P,O);	*pasosp*	*oasosp*	P	-	-	
T						*0*	*0*	-			
F		rP				*parpsp*	*oarpsp*	P	O		O failed to support a claim it agreed with
T						*1*	*-1*	-			
F	a	sO		rO		*pasoro*	*oasoro*	P	-	P	P failed to support a claim it proposed
T						*-1*	*1*	-			
F		rP				*parpro*	*oarpro*	P	O		both above
T						*-1*	*-1*	-			

Figure 9: Improved Agreement



claim	dec	out	predicate	P	O	fB	nB	oB	oB Blame Justification
F	a	sO	refute(c,O,P); p(c,I,S)? (paso,oaso): (parp,oarp)	*paso*	*oaso*	P(8)		-	
T				*0*	*0*	-	P(15)		
T-Opt						-	-		
F		rP		*parp*	*oarp*	P(8)		O	O failed to support a claim it agreed with
T				*1*	*-1*	-	P(15)		
T-Opt							-		
F	d	rO	refute(c,P,O); p(c,I,S)? (pdsp,odsp): (pdro,odro)	*pdro*	*odro*	P(8)		P	P failed to support a claim it proposed
T				*-1*	*1*	-	P(15)		
T-Opt							-		
F		sP		*pdsp*	*odsp*	P(8)		O	O did not refute a claim it disputed
T				*1*	*-1*	-	P(15)		
T-Opt							-		
F	s(c')	sO	refute(c',O,P); p(c',I,S)? (psso,osso): (psrp,osrp)	*psso*	*osso*	P(8)		P	P failed to refute a stronger claim than the claim it proposed
T				*-1*	*1*	-	P(15)		
T-Opt							-		
F		rP		*psrp*	*osrp*	P(8)		O	O failed to support the strengthened claim
T				*1*	*-1*	-	P(15)		
T-Opt							-		

Figure 10: Complete Table for Optimization

strategy and in a second try you might fail to refute.

15. MEASURE LEARNING

15.1 Student Assessment with SCG

SCG has an natural assessment approach implied by the Scientific Method.

15.1.1 A perfect master teacher is available

input: claim; output: true, false, optimal

input: true claim; output: instance that leads to defense

input: true claim, instance; output: does instance lead to defense?

input: true claim, instance; output: solution that defends claim

MAKE GENERIC

input: false claim. output: first step in refute(c,P,O) that leads to refutation.

input: false claim. Partial elaboration of refute(c,P,O) with next step to be made by O. output: step by O that leads to refutation.

input: true claim. Partial elaboration of refute(c,P,O) with next step to be made by P. output: step by P that leads to defense.

The above perfect master teacher capabilities can be used to guide and assess the student.

15.1.2 No perfect master teacher

We still have the blame assigned based on the refutation protocol outcome (oB column in Figure 8.

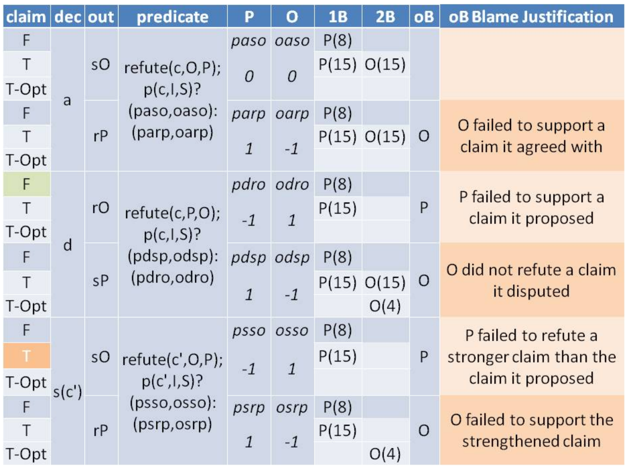
reason for loss (e.g., proposed claim refutation) not easy to find claim could be false and properly attacked (error in propose) claim could be false and improperly attacked and improperly defended (error in propose,provide and solve) claim could be true but not properly defended (error in provide or solve)

don't know in which situation we are. How does SCG help?

Yes, SCG helps: reason: (oB column in Figure 8.

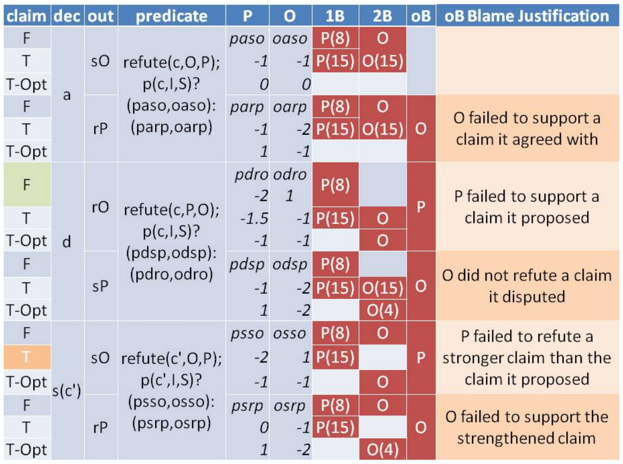
15.2 Learning Science and SCG

I understand your concerns about incorporating learning scientists. I believe, SCG has very good learning science built in. Below is a description how learning happens and how it is measured in



claim	dec	out	predicate	P	O	1B	2B	oB	oB Blame Justification
F	a	sO	refute(c,O,P); p(c,I,S)? (paso,oaso): (parp,oarp)	paso	oaso	P(8)			
T				0	0	P(15)	O(15)		
T-Opt									
F		rP		parp	oarp	P(8)		O	O failed to support a claim it agreed with
T				1	-1	P(15)	O(15)		
T-Opt									
F	d	rO	refute(c,P,O); p(c,I,S)? (pdsp,odsp): (pdro,odro)	pdro	odro	P(8)		P	P failed to support a claim it proposed
T				-1	1	P(15)			
T-Opt									
F		sP		pdsp	odsp	P(8)		O	O did not refute a claim it disputed
T				1	-1	P(15)	O(15)		
T-Opt							O(4)		
F	s(c')	sO	refute(c',O,P); p(c',I,S)? (psso,osso): (psrp,osrp)	psso	osso	P(8)		P	P failed to refute a stronger claim than the claim it proposed
T				-1	1	P(15)			
T-Opt									
F		rP		psrp	osrp	P(8)		O	O failed to support the strengthened claim
T				1	-1	P(15)			
T-Opt							O(4)		

Figure 11: All Learning Opportunities



claim	dec	out	predicate	P	O	1B	2B	oB	oB Blame Justification
F	a	sO	refute(c,O,P); p(c,I,S)? (paso,oaso): (parp,oarp)	paso	oaso	P(8)	O		
T				-1	-1	P(15)	O(15)		
T-Opt				0	0				
F		rP		parp	oarp	P(8)	O	O	O failed to support a claim it agreed with
T				-1	-2	P(15)	O(15)		
T-Opt				1	-1				
F	d	rO	refute(c,P,O); p(c,I,S)? (pdsp,odsp): (pdro,odro)	pdro -2	odro 1	P(8)		P	P failed to support a claim it proposed
T				-1.5	-1	P(15)	O		
T-Opt				-1	-1		O		
F		sP		pdsp	odsp	P(8)		O	O did not refute a claim it disputed
T				-1	-2	P(15)	O(15)		
T-Opt				1	-2		O(4)		
F	s(c')	sO	refute(c',O,P); p(c',I,S)? (psso,osso): (psrp,osrp)	psso	osso	P(8)	O	P	P failed to refute a stronger claim than the claim it proposed
T				-2	1	P(15)			
T-Opt				-1	-1		O		
F		rP		psrp	osrp	P(8)	O	O	O failed to support the strengthened claim
T				0	-1	P(15)			
T-Opt				1	-2		O(4)		

Figure 12: With Master Scholar

SCG.

In an SCG lab, learning happens during the elaboration of the refutation protocol for a claim. When a claim is defended or refuted, there is a sequence S of instances and solutions which has been produced by the refutation protocol. If the claim is defended, the claim predicate evaluates to true for S. The sequence S contains a surprise for the opponent of the claim because the opponent's intention was to make the predicate false. This surprise is the crystallization point for learning. The student playing the role of the opponent is encouraged to ask and answer the following questions: (O1) Why is my prediction wrong that I will successfully refute? (O2) What is the general pattern behind the clever construction that my partner used to defend the claim? Can I interfere with the clever construction? Can I reconstruct it from S? (O3) Can I defend the claim against a partner, successfully? (O4) Can I improve my approach to trying to refute the claim in a second attempt? (O5) Do I still believe that I can refute the claim? (O6) Did I make a mistake? Was there a second or third mistake? Do a blame assignment.

The proponent of the claim is pleased with winning but is not off the hook: (P1) Did I win by accident? Has the opponent made a mistake which made me win this time but not against a better partner? (P2) How do I repeat my success even when the opponent plays differently? (P3) Have I a systematic defense strategy? (P4) Works my systematic defense strategy in all cases?

Emotions of the proponent when she wins: joy, I found a clever construction to defend. Emotions of the opponent when he loses: disappointment, I will try to figure out your clever construction and maybe change my mind about trying to refute.

SCG offers the following approach to measure learning in a lab for a given student: [unsuccessful => successful] Defense attempts are unsuccessful (dau)=> defense attempts are successful (das). Student learned to recognize, correctly, defensible claims. Refutation attempts are unsuccessful (rau) => refutation attempts are successful (ras). Student learned to recognize, correctly, refutable claims. Agreement attempts are unsuccessful (aau) => agreement attempts are successful (aas). Student learned to recognize, correctly, optimal claims. Amount learned: dau-das + rau-ras + aau-aas

[change of mind] Claim C was unsuccessfully defended => claim C is successfully refuted consistently Claim C was unsuccessfully refuted => claim C is successfully defended consistently Amount learned: number of claims where a change of mind happened.

16. SMALL LABS

Labs with c and !c.

17. LABS WITH PERFECT AVATARS

Useful for learning. Always have perfect answers. But costly to produce.

18. INTERESTING PAYOFF FUNCTIONS

Looking at Figure 8, there are two blame justifications where O did not do anything wrong. It would be natural to give a higher payoff to O in these two cases: odro = 2, osso = 2.

If O is blamed in oB, P might also have contributed misinformation: P might have proposed a false claim. It makes sense to give a lower payoff to P: parp = 1, pdsp = 1, psrp = 1.

19. PROBLEM SOLVING

An important goal of the SCG is to make the learners better problem solvers. The problems to be solved: Find good claims (true or optimal claims) and find good provideInstance and solveInstance functions.

Lab Reductions are a useful tool in this process. Lab L1 is a reduction of lab L2 if a winning strategy for L1 implies a winning strategy for L2.

======from slides

With the next example we show the usefulness of lab reductions. A lab L1 reduces to a lab L2 (L1 < L2) if a defense strategy for the claims in L2 guarantees a defense strategy for the claims in L1. Ideally, the claims in L2 are simpler.

L1 reduces to L2 if we can use a black box for L2 to solve L1. The black box makes all perfect decisions, including claims it can defend.

A mapping from L1 to L2 is a computable function f Domain Claim such that for any L1.Domain L2.Domain L1.Claim L2.Claim propose oppose/agree provideInstance solveInstance refute

=========

Incremental approach A successful refutation of claim c is viewed as a small step towards a proof of the negation of c. If the proponent is perfect, the successful refutation counts as a proof of !c because the perfect proponent would have found a way to defend if such a defense of !c exists.

A successful defense of claim c is viewed as a small step towards a proof of c. If the opponent is perfect, the successful defense counts as a proof of c because the perfect opponent would have found a way to refute if such a refutation of c exists. Restriction: if the opponent is not perfect, it is possible that c is false and the defense happened because the opponent made a mistake.

19.1 Convergence

When no blame is assigned during a binary game in an optimization lab, the optimum claim will eventually be found.

Theorem [Convergence]: Consider a set C of claims $c(t)$, where t is a real number between 0 and 1. $c(0)$ is true, and $c(1)$ is false and there is an optimal value t_0 of t where the truth value of $c(t_0)$ switches from true to false. If a sequence of binary games is played using claims in C and binary search without faulty actions, the optimal claim $c(t_0)$ will be found.

19.2 Indeterminate Claims

SCG can express indeterminate claims that are neither true nor false. Such claims were studied in Independence Friendly Logic [34], an extension of first-order logic.

Consider the following lab: *Instance* = the set of positive real numbers = *InstanceSet*. *Solution* = the set of real numbers. The $valid(i, s)$ function checks that the solution s is the square root of instance i. The protocol is: $P : i[0], O : s[1]$ of $i[0], P : s[2]$ of $i[0]$. The protocol predicate is: $s[1] = s[2]$. According to the SCG rules, $s[1]$ is not known when $s[2]$ is computed. The lab contains only one claim which is neither true nor false: it is indeterminate. Notice the similarity to the "at least as good as" claim discussed earlier.

Theorem [ExistIndeterminateClaims]: There are indeterminate SCG claims.

20. ACCIDENTAL DEFENSES

20.1 Avoidable Accidental Defenses

Detected by game rules. Instance must be in instance set. solution must be valid.

False claim would be defended because Bob is careless. Bob is kicked.

20.2 Skill-related Accidental Defenses

21. PROBLEM SOLVING COURSES

http://www.ccs.neu.edu/home/lieber/evergreen/specker/SCG-Teach/teach.html

22. RULES FOR REPUTATION COMPUTATION

We have developed a computational model for scientific communities to foster better innovation and better education. Central to a scientific community is refutation and how it affects reputation of the scholars. The following rules define the reputation mechaism of SCG.

There is some redundancy in those rules but I believe no contradiction.

Scholars propose and oppose claims and agree on claims. Oppose means (refute | strengthen). Refute is determined by a refutation protocol. Strengthening is reduced to refutation. Agreement is also reduced to refutation.

Strengthening: When claim C is strengthened by Bob to C', Alice must try to refute C' and the strengthening holds only if Bob defends C'. strengthenP(C,C') must hold. When scholar Bob successfully strengthens a claim of Alice, Bob wins reputation: Bob + ClaimConfidence + |quality(C)-quality(C')| When scholar Alice successfully defends her own claim against Bob, Alice wins reputation. Alice + ClaimConfidence

There is a gray zone with strengthening. Let's assume we have quality(C) < q < quality(C') and q is the quality achieved by the solution. Then both Alice and Bob have lost because Bob did not achieve what he claimed and Alice claim was shown not to be optimal. We make the simplifying assumption that Bob only wins if he defends C'.

Agreement: When Bob agrees on claim C with Alice, (1) Bob must defend C against Alice (if not, Bob loses) (2) Bob must refute C' = C minimally strengthened along quality dimension (using the configuration file constant minStrengthen) with Alice as defender (if not, Bob loses). Then Alice must do the same: (1) Alice must defend C against Bob (if not, Alice loses) (2) Alice must refute C' with Bob as defender (if not, Alice loses) If all those protocols produce the result as described, the claim goes into the social welfare set (the knwledge base of claims believed to hold and having maximum strength).

All scholars start with reputation 100. Reputation is zero sum. Alice proposes, Bob opposes.

When scholar Bob successfully refutes a claim of Alice, Bob wins reputation: Bob + ClaimConfidence

When scholar Alice successfully defends her own claim against Bob, Alice wins reputation. Alice + ClaimConfidence

summary: Bob: + ClaimConfidence * result Alice: - ClaimConfidence * result

When scholar Bob successfully strengthens a claim C of scholar Alice to claim C', Bob wins reputation: Bob + ClaimConfidence + |quality(C)-quality(C')|

Checking of instances and solutions:

0. An InstanceSet must be valid. 1. All instances are in Instance. 2. A solution s in Solution for instance i in Instance must satisfy: valid(i,s). 3. When an instance i in Instance is provided, InstanceSet.belongsTo(i) holds.

In one domain, multiple InstanceSet are allowed. In one play ground, multiple claims are allowed.

Some rules are enforced syntactically by the structure of a game definition. Only one domain definition. Multiple different claim languages are allowed, e.g., claims and negated claims.

Avatars with a negative reputation are kicked from the game.

The constants in the configuration file are enforced.

Axioms

Scholars gain reputation either by opposing (refuting or strengthening) other scholars' claims or by having their claims defended against other agents. Scholar's gain from their claim is proportional to both the confidence of their claim and the result of the refutation protocol (a value in [-1,1]).

One scholar's reputation gain is another scholar's reputation loss. The sum of all agent's reputation is preserved.

Arguments (instances and solutions obtained from the refutation protocol) recognize claims by a recognition factor in [-1,1]. A recognition factor of 1 means that the other scholar Bob has completely failed to discount Alice' claim. We say that Alice has defended the claim. A recognition factor of -1 means that the other scholar has completely succeeded to refute the claim. We say that Bob has successfully refuted the claim.

Claims have a confidence in [0,1].

The scholar's confidence reflects the amount of effort made by the scholar to refute the claim. If it is a mathematical claim, it is the amount of effort spent to try to prove the claim (i.e. turning it into a theorem). Scholar's reputation is the accumulation of the scholar's initial reputation and its reputation gains and losses; thus reflecting the past performance of the scholar.

Those axioms define a family of mechanisms that can be used to implement the game.

23. EXAMPLE

Homework 3 Algorithms and Data Spring 2012 Karl Lieberherr

Due date: Feb. 2, 2012, beginning of class.

Read Chapter 3 in the text book. By now you should have covered chapters 1 through 3.

We are going to put the proposer of a claim into the claim: claim XYZ(Name, ...) where Name is the name of the team, e.g., Griffin-Schneider+Christopher-Souvey or if you work by yourself, Kevin-Castaglia.

PART 1: Proposed by Ahmed Abdelmeged =======

In this homework, we study an existing algorithm, the Gale-Shapley algorithm, and we want to find out how slow or how fast it runs depending on the input.

Given an algorithm A:X -> Y and some input size n, our goal is to find the worst input x so that some resource function: A-resource(x): X -> PositiveRational is maximum over all inputs of the same size n. Below we consider a decision variant of this optimization question.

We consider claims of this form: Given an algorithm A: X -> Y and an input size n, there exists an input x of size n so that A-resource(x) is >= c. A-resource is defined by an instrumentation of the algorithm and we assume that it returns a value in [0,1]. We abbreviate this claim as MAX-RES(Name,A,n,c). Similarly, we define claim MIN-RES(Name,A,n,c).

Example: A = Gale-Shapley: Gale-Shapley-resource(p) is

```
the number of iterations of the while loop  for preference p / n^2,
```

where n is the number of men = number of women. Gale-Shapley-resource(p) is a rational number between 0 and 1.

We define the JSON notation for defining a preference p as follows:

"n":3, "manPref" : [[2,1,0],[1,0,2],[0,1,2]], "womanPref : [[2,1,0],[1,3,2],[3,1,2]]

This notation is matching Ahmed's Java program presented in class and here:

http://www.ccs.neu.edu/home/lieber/courses/algorithms/cs4800/sp12/lectures/GaleShapley

Claims are of the form: MAX-RES(Name, Gale-Shapley, n, 0.8) or MIN-RES(Name,Gale-Shapley, n, 0.1), where n is the number of men = number of women and Name is the student/team name. What are the optimum claims? About 5 teams should post an optimum claim on Piazza. When a claim is challenged, the preference (i.e., the input) must be given. Each proposed claim on Piazza must be either agreed, refuted or strengthened.

What to turn in: The protocols of the quantifier games you played with your partner. A description of your approach to find optimum claims and a description of your defense strategy for your optimum claims.

PART 2:

This homework part is about determining the asymptotic behavior of the functions we computed in hw 2: HSR(n,k) = q and M(k,q) = n. We define HSR(n,k) to be the smallest number of questions needed in the worst-case for a ladder with rungs 0..n-1 and a jar budget of k. M(k,q) is the maximum number of rungs we can handle with k jars to break and q questions.

We play again the quantifier game.

The scholars make claims of the form:

Landau(Name, HSR(n,k), O(exp)) meaning HSR(n,k) in O(exp).

Landau(Name, M(k,q), O(exp)) meaning M(k,q) in O(exp).

Landau(Name, NOT, HSR(n,k), O(exp)) meaning HSR(n,k) !in O(exp) (negative claim)

Landau(Name, NOT, M(k,q), O(exp)) meaning M(k,q) !in O(exp)

where exp is an expression using powers (including fractional exponents), logarithms and exponential functions.

The same for Big Omega and Big Theta in addition to Big O.

```
Example claims:
HSR(n,2) in O(n^(1/2))
or
Landau(Karl,HSR(n,2), O(n^(1/2)))
```

HSR(n,2) in O(n) Landau(Karl,HSR(n,2), O(n))

HSR(n,2) in O(n) or Landau(Karl,HSR(n,2),O(n))

What to turn in:

1. Game history: List all claims proposed, refuted and strengthened in the order they happened in the quantifier game with your partner. The class should put about 5 claims on Piazza to illustrate how refutations and defenses work in this case.

2. Your asymptotic bounds for HSR(n,k) and M(k,q).

24. RELATED WORK

The SCG has not grown in a vacuum. We make connections to several related areas.

24.1 ToDo

Paper by Sebastian Deterding: From Game Design Elements to Gamefulness: Defining Gamification. MindTrek 11, ACM. Augmented reality games that use digital devices to overlay game representations over the environment [50].

From Wikipedia: Education and AR:

Augmented reality applications can complement a standard curriculum. Text, graphics, video and audio can be superimposed into a studentÂŠs real time environment. Textbooks, flashcards and other educational reading material can contain embedded ÂŞmarkersÂŤ that, when scanned by an AR device, produce supplementary information to the student rendered in a multimedia format.[59][60][61] Students can participate interactively with computer generated simulations of historical events, exploring and learning details of each significant area of the event site.[62] AR can aide students in understanding chemistry by allowing them to visualize the spatial structure of a molecule and interact with a virtual model of it that appears, in a camera image, positioned at a marker held in their hand.[63] Augmented reality technology also permits learning via remote collaboration, in which students and instructors not at the same physical location can share a common virtual learning environment populated by virtual objects and learning materials and interact with another within that setting.[64]

[64] Collaborative Augmented Reality in Education by Hannes Kaufmann

Connection between augmented reality and SCG. For learning: Students pose problems to each other and solve them. Structured scientific discourse.

intrinsically motivating instruction by Tom Malone http://mailer.fsu.edu/ jkeller/I

which book by Dan Pink should we reference? If what he says is right, SCG will be a big thing when little or no monetary rewards are offered.

Dan Pink's three principles Purpose, Mastery, and Autonomy. I think there could be several ways to map these three principles onto SCG. Here is my shot:

Mastery : of knowledge about a particular problem solving domain (i.e. how to find (good) solutions to problem instances, what are the hard instances?) Mastery is manifested by the ability to provide harder to falsify claims about players ability to solve problem instances as well as the ability to spot problems in other players claims.

Autonomy : Players are free to choose the claims they propose. There are several restrictions on autonomy imposed by the game as well. For example, players don't choose the claims they want to dispute. Players do not choose their action time. Claims proposed by the players are restricted by the lab designer.

BUT: the players choose the lab they want to play in (out of thousands of labs).

Also players should have a way to interact with lab designers to propose modified labs.

=================

SCHECHTER, S. E. How to buy better testing: using competition to get the most security and robustness for your dollar

BACON, D., CHEN, Y., PARKES, D., AND RAO, M. A market-based approach to software evolution. OOPSLA ÃĆ9: Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications (2009).

24.2 Crowd Sourcing and Human Computation

There are several websites that organize competitions. What is common to many of those competitions? We believe that the SCG provides a foundation to websites such as TopCoder.com or kaggle.com.

The SCG makes a specific, but incomplete proposal of a programming interface to work with the global brain [6]. What is currently missing is a payment mechanism for scholars and an algorithm to split workers into pairs based on their background.

The SCG is a generic version of the "Beat the Machine" approach for improving the performance of machine learning systems [4].

Scientific discovery games, such as FoldIt and EteRNA, are variants of the SCG. [8] describes the challenges behind developing scientific discovery games. [3] argues that complex games such as FoldIt benefit from tutorials. This also applies to the SCG, but a big part of the tutorial is reusable across scientific disciplines.

24.3 Logic and Imperfect Information Games

Logic has long promoted the view that finding a proof for a claim is the same as finding a defense strategy for a claim.

Logical Games [26], [12] have a long history going back to Socrates. The SCG is an imperfect information game which builds on Paul Lorenzen's dialogical games [17].

24.4 Foundations of Digital Games

A functioning game should be deep, fair and interesting which requires careful and time-consuming balancing. [14] describes techniques used for balancing that complement the expensive playtesting. This research is relevant to SCG lab design. For example, if there is an easy way to refute claims without doing the hard work, the lab is unbalanced.

24.5 Architecting Socio-Technical Ecosystems

This area has been studied by James Herbsleb and the Center on Architecting Socio-Technical Ecosystems (COASTE) at CMU http://www.coaste.org/. A socio-technical ecosystem supports straightforward integration of contributions from many participants and allows easy configuration.

The SCG has this property and provides a specific architecture for building knowledge bases in (formal) sciences. Collaboration between scholars is achieved through the scientific discourse which exchanges instances and solutions. The structure of those instances and solutions gives hints about the solution approach. An interesting question is why this indirect communication approach works.

The NSF workshop report [30] discusses socio-technical innovation through future games and virtual worlds. The SCG is mentioned as an approach to make the scientific method in the spirit of Karl Popper available to CGVW (Computer Games and Virtual Worlds).

24.6 Online Judges

An online judge is an online system to test programs in programming contests. A recent entry is [28] where private inputs are used to test the programs. Topcoder.com includes an online judge capability, but where the inputs are provided by competitors. This dynamic benchmark capability is also expressible with the SCG: The claims say that for a given program, all inputs create the correct output. A refutation is an input which creates the wrong result.

24.7 Educational Games

The SCG can be used as an educational game. One way to create adaptivity for learning is to create an avatar that gradually poses harder claims and instances. Another way is to pair the learner with another learner who is stronger. [2] uses concept maps to guide the learning. Concept maps are important during lab design: they describe the concepts that need to be mastered by the students for succeeding in the game.

24.8 Formal Sciences and Karl Popper

James Franklin points out in [11] that there are also experiments in the formal sciences. One of them is the 'numerical experiment' which is used when the mathematical model is hard to solve. For example, the Riemann Hypothesis and other conjectures have resisted proof and are studied by collecting numerical evidence by computer. In the SCG experiments are performed when the refutation protocol is elaborated.

Karl Popper's work on falsification [29] is the father of non-deductive methods in science. The SCG is a way of doing science on the web according to Karl Popper.

24.9 Scientific Method in CS

Peter Denning defines CS as the science of information processes and their interactions with the world [9]. The SCG makes the scientific method easily accessible by expressing the hypotheses as claims. Robert Sedgewick in [31] stresses the importance of the scientific method in understanding program behavior. With the SCG, we can define labs that explore the fastest practical algorithms for a specific algorithmic problem.

24.10 Games and Learning

Kevin Zollman studies the proper arrangement of communities of learners in his dissertation on network epistemology [35]. He studies the effect of social structure on the reliability of learners.

In the study of learning and games the focus has been on learning known, but hidden facts. The SCG is about learning unknown facts, namely new constructions.

24.11 CSP-based Game Design

CSP is increasingly being used in the procedural content generation (PCG) community, although not in industry. For example, Tanagra[32] uses a numerical constraint solver to guarantee level playability. In addition, Magy El-Nasr used constraint solving for lighting and adaptive systems for games [10].

24.12 Origins of SCG

A preliminary definition of the SCG was given in a keynote paper [23]. [21] gives further information on the Scientific Community Game. The original motivation for the SCG came from the two papers with Ernst Specker: [24] and the follow-on paper [25]. Renaissance competitions are another motivation: the public problem solving duel between Fior and Tartaglia, about 1535, can easily be expressed with the SCG protocol language.

25. FUTURE WORK

We see a significant potential in putting the refutation-based Scientific Method into the cyberinfrastructure and make it widely available. We plan to, iteratively, improve our current implementation based on user feedback.

We see an interesting opportunity to mine the game histories and make suggestions to the scholars how to improve their skills to propose and defend claims. If this approach is successful, the SCG will make contributions to computer-assisted problem solving.

26. SUMMARY AND CONCLUSIONS

The SCG provides a simple interface to a community that uses the (Popperian) Scientific Method. The SCG provides for effective customization of the generic scientific machinery by using lab definitions. Since the SCG models a scientific community it is a broad enabling tool for innovation and learning and deserves a central place in the world's cyberinfrastructure and serious games world. We believe that the game design approach we outline in this paper has many applications to other games. We start with a game goal and translate it into a blame assignment for moves that are inconsistent with the design goal. Then we derive a payoff function that is fair, sound and competitive. Such a systematic approach eliminates a lot of game testing because we know that many properties are formally guaranteed.

Acknowledgments: We would like to thank Bryan Chadwick, Magy Seif El-Nasr, David Lazer, Rory Smead, Abraham Bernstein and Gillian Smith for their input and feedback on the paper.

27. EXPERIENCE WITH THE SCG

The SCG has evolved since 2007. We have used the SCG in software development courses at both the undergraduate and graduate level and in several algorithm courses. Detailed information about those courses is available from the second author's teaching page.

27.1 Software Development

The most successful graduate classes were the ones that developed and maintained the software for SCG Court [1] as well as several labs and their avatars to test SCG Court. Developing labs for avatars has the flavor of defining a virtual world for artificial creatures. At the same time, the students got detailed knowledge of some problem domain and how to solve it. A fun lab was the Highest Safe Rung lab from [19] where the best avatars needed to solve a constrained search problem using a modified Pascal triangle.

27.2 Algorithms

The most successful course (using [19] as textbook) was in Spring 2012 where the interaction through the SCG encouraged the students to solve difficult problems. Almost all homework problems were defined through labs and the students posted both their exploratory and performatory actions on piazza.com. We used a multiplayer version of the SCG binary game which created a bit of an information overload. Sticking to binary games would have been better but requires splitting the students into pairs. The informal use of the SCG through Piazza (piazza.com) proved successful. All actions were expressed in JSON which allowed the students to use a wide variety of programming languages to implement their algorithms.

The students collaboratively solved several problems such as the problem of finding the worst-case inputs for the Gale-Shapley algorithm (see the section Example above).

We do not believe that, without the SCG, the students would have created the same impressive results. The SCG effectively focuses the scientific discourse on the problem to be solved.

The SCG proved to be adaptive to the skills of the students. A few good students in a class become effective teachers for the rest thanks to the SCG mechanism.

28. RELATED WORK

The SCG has not grown in a vacuum. We make connections to several related areas.

28.1 Crowd Sourcing and Human Computation

28.1.1 Dealing with Unreliable Workers

Most crowdsourcing systems must devise schemes to increase confidence in the worker's solutions to tasks, typically by assigning each task multiple times [16]. Karger et al. present a general model for crowdsourcing tasks. In SCG, because workers need to justify their answers in a game against another worker, unreliable workers will run into many contradictions and get a low rating. This means that their votes will minimally affect the final result, the knowledge base of true claims.

[7] is related to SCG scholar ranking. The algorithm is an extended Bradley-Terry model called Crowd-BT. The paper focuses on finding the quality of annotators in a crowdsourced setting. They study the exploration-exploitation tradeoff which is also relevant to SCG for labeling claims as true or false.

The "Evaluating the Crowd with Confidence" paper [15] has a title that seems very applicable to SCG. However, they use a model which is too simple for SCG. In particular, in SCG the errors depend on task difficulty, and worker erros are not independent of each other because they play a game.

28.1.2 Rating Systems

We use a rating system for games with wins, losses and draws. This subject has been studied for a long time and there are many applications of rating systems. For example, in chess and other sports, the Elo rating system is used. A good survey and critique of rating systems is given in [5]. Rating systems are a controversial subject and there are many algorithms that can be used. TopCoder [33] uses an Algorithm Competition Rating System to rank the coders.

28.1.3 Combining Worker's Contributions

In SCG, we use two approaches to combine scholar contributions: (1) During the refutation games, the scholars give each other feedback by trying to drive each other into a contradiction. This is a collaboration which leads potentially to new ideas and knowledge fusion. (2) We combine the votes with justifications into an overall vote for whether a claim is true. Related work is [7] and [16] which was already discussed above.

28.1.4 Competitions

There are several websites that organize competitions. What is common to many of those competitions? We believe that the SCG provides a foundation to websites such as TopCoder.com or kaggle.com.

The SCG makes a specific, but incomplete proposal of a programming interface to work with the global brain [6]. What is currently missing is a payment mechanism for scholars and an algorithm to split workers into pairs based on their background.

The SCG is a generic version of the "Beat the Machine" approach for improving the performance of machine learning systems [4].

Scientific discovery games, such as FoldIt and EteRNA, are variants of the SCG. [8] describes the challenges behind developing scientific discovery games. [3] argues that complex games such as FoldIt benefit from tutorials. This also applies to the SCG, but a big part of the tutorial is reusable across scientific disciplines.

28.1.5 Crowdsourcing complex tasks

[18] describes a general-purpose framework for solving complex problems through micro-task markets. Engaging in the scientific dialogs of FSCP could be done through a micro-task market. [27] proposes a language to define crowdsourcing systems. Our lab definition approach provides a declarative description of what needs to be crowdsourced.

[20] provides an interesting analysis of several issues relevant to FSCP: how incorrect responses should affect worker reputations and how higher reputation leads to better results.

28.2 Logic and Imperfect Information Games

Logic has long promoted the view that finding a proof for a claim is the same as finding a defense strategy for a claim.

Logical Games [26], [12] have a long history going back to Socrates. The SCG is an imperfect information game which builds on Paul Lorenzen's dialogical games [17].

28.3 Foundations of Digital Games

A functioning game should be deep, fair and interesting which requires careful and time-consuming balancing. [14] describes techniques used for balancing that complement the expensive playtesting. This research is relevant to SCG lab design. For example, if there is an easy way to refute claims without doing the hard work, the lab is unbalanced.

28.4 Architecting Socio-Technical Ecosystems

This area has been studied by James Herbsleb and the Center on Architecting Socio-Technical Ecosystems (COASTE) at CMU http://www.coaste.org/. A socio-technical ecosystem supports straightforward integration of contributions from many participants and allows easy configuration.

The SCG has this property and provides a specific architecture for building knowledge bases in (formal) sciences. Collaboration between scholars is achieved through the scientific discourse implied by the refutation game. The information exchanged gives hints about how to play the game better next time. An interesting question is why this indirect communication approach works.

The NSF workshop report [30] discusses socio-technical innovation through future games and virtual worlds. The SCG is mentioned as an approach to make the scientific method in the spirit of Karl Popper available to CGVW (Computer Games and Virtual Worlds).

28.5 Online Judges

An online judge is an online system to test programs in programming contests. A recent entry is [28] where private inputs are used to test the programs. Topcoder.com [33] includes an online judge capability, but where the inputs are provided by competitors. This dynamic benchmark capability is also expressible with the SCG: The claims say that for a given program, all inputs create the correct output. A refutation is an input which creates the wrong result.

28.6 Educational Games

The SCG can be used as an educational game. One way to create adaptivity for learning is to create an avatar that gradually poses harder claims and makes the scientific discourse more challenging. Another way is to pair the learner with another learner who is stronger. [2] uses concept maps to guide the learning. Concept maps are important during lab design: they describe the concepts that need to be mastered by the students for succeeding in the game.

28.7 Formal Sciences and Karl Popper

James Franklin points out in [11] that there are also experiments in the formal sciences. One of them is the 'numerical experiment' which is used when the mathematical model is hard to solve. For example, the Riemann Hypothesis and other conjectures have resisted proof and are studied by collecting numerical evidence by computer. In the SCG experiments are performed when the game associated with a claim is elaborated.

Karl Popper's work on falsification [29] is the father of non-deductive methods in science. The SCG is a way of doing science on the web according to Karl Popper.

28.8 Scientific Method in CS

Peter Denning defines CS as the science of information processes and their interactions with the world [9]. The SCG makes the scientific method easily accessible by expressing the hypotheses as claims. Robert Sedgewick in [31] stresses the importance of the scientific method in understanding program behavior. With the SCG, we can define labs that explore the fastest practical algorithms for a specific algorithmic problem.

28.9 Games and Learning

Kevin Zollman studies the proper arrangement of communities of learners in his dissertation on network epistemology [35]. He studies the effect of social structure on the reliability of learners.

In the study of learning and games the focus has been on learning known, but hidden facts. The SCG is about learning unknown facts, namely new constructions.

28.10 CSP-based Game Design

CSP is increasingly being used in the procedural content generation (PCG) community, although not in industry. For example, Tanagra[32] uses a numerical constraint solver to guarantee level playability. In addition, Magy El-Nasr used constraint solving for lighting and adaptive systems for games [10].

28.11 Origins of SCG

A preliminary definition of the SCG was given in a keynote paper [23]. [21] gives further information on the Scientific Community Game. The original motivation for the SCG came from the two papers with Ernst Specker: [24] and the follow-on paper [25].

[22] describes an earlier version of SCG. The key difference is that the old SCG was targeted at evalauation of the scholars while FSCP is targeted at crowdsourcing true claims. FSCP is cleaner: there is a simple concept of self-contradiction and there is no longer a need to have the concept of strengthening a claim explicitly.

29. FUTURE WORK

We want to extend our model so that we can make claims about claims. For example, we want to have a "macro" for a claim to be optimal. We want to leverage claim relationships across labs and work with lab reductions as a useful problem solving tool.

We see a significant potential in putting the refutation-based Scientific Method into the cyberinfrastructure and make it widely available. We plan to, iteratively, improve our current implementation based on user feedback.

We see an interesting opportunity to mine the game histories and make suggestions to the scholars how to improve their skills to propose and defend claims. If this approach is successful, the SCG will make contributions to computer-assisted problem solving.

30. CONCLUSIONS

The SCG provides a simple interface to a community that uses the (Popperian) Scientific Method. The SCG provides for effective customization of the generic scientific machinery by using lab definitions. Since the SCG models a scientific community it is a broad enabling tool for innovation and learning and deserves a central place in the world's cyberinfrastructure and serious games world.

31. ABSTRACT

Crowdsourcing contests have received a lot of attention in recent years. We study the general problem how to use crowdsourcing to build knowledge bases and to collect the know-how to defend the claims in the knowledge base. We express claims in a knowledge base as predicate logic formulas. To challenge the crowd and discourage weak participants, all assertions of the form: "This claim is true" must be substantiated by one or more games to be won. All these substantiation games are refutation games associated with the formula of the claim.

We define a generator of crowdsourcing systems which is paramaterized by labs that focus the crowd on a specific task. We mention key properties of our system and we report on our experience in using the approach in teaching.

32. ACKNOWLEDGEMENTS

We would like to thank Magy Seif El-Nasr, Casper Harteveld, Thomas Wahl and Tugba Koc for their input to the paper.

33. REFERENCES

[1] A. Abdelmeged and K. J. Lieberherr. SCG Court: Generator of teaching/innovation labs on the web. Website, 2011. `http://sourceforge.net/p/generic-scg/code-0/110/tree/GenericSCG/` .

[2] E. Andersen. Optimizing adaptivity in educational games. In *Proceedings of the International Conference on the Foundations of Digital Games*, FDG '12, pages 279–281, New York, NY, USA, 2012. ACM.

[3] E. Andersen, E. O'Rourke, Y.-E. Liu, R. Snider, J. Lowdermilk, D. Truong, S. Cooper, and Z. Popovic. The impact of tutorials on games of varying complexity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 59–68, New York, NY, USA, 2012. ACM.

[4] J. Attenberg, P. Ipeirotis, and F. Provost. Beat the machine: Challenging workers to find the unknown unknowns. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[5] J. Beasley. *The Mathematics of Games*. Dover Books on Mathematics. Dover Publications, 2006.

[6] A. Bernstein, M. Klein, and T. W. Malone. Programming the global brain. *Commun. ACM*, 55(5):41–43, May 2012.

[7] X. Chen, P. N. Bennett, K. Collins-Thompson, and E. Horvitz. Pairwise ranking aggregation in a crowdsourced setting. In *WSDM, Rome, Italy*, 2013.

[8] S. Cooper, A. Treuille, J. Barbero, A. Leaver-Fay, K. Tuite, F. Khatib, A. C. Snyder, M. Beenen, D. Salesin, D. Baker, and Z. Popović. The challenge of designing scientific discovery games. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 40–47, New York, NY, USA, 2010. ACM.

[9] P. J. Denning. Is computer science science? *Commun. ACM*, 48(4):27–31, Apr. 2005.

[10] M. S. El-Nasr and I. Horswill. Automating lighting design for interactive entertainment. *Comput. Entertain.*, 2(2):15–15, Apr. 2004.

[11] J. Franklin. The formal sciences discover the philosophers' stone. *Studies in History and Philosophy of Science*, 25(4):513–533, 1994.

[12] W. Hodges. Logic and games. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2009 edition, 2009.

[13] P. Ipeirotis, F. Provost, V. Sheng, and J. Wang. Repeated labeling using multiple noisy labelers. *This work was supported by the National Science Foundation under GrantNo. IIS-0643846, by an NSERC P, Vol*, 2010.

[14] A. Jaffe, A. Miller, E. Andersen, Y.-E. Liu, A. Karlin, and Z. Popovic. Evaluating competitive game balance with restricted play, 2012.

[15] M. Joglekar, H. Garcia-Molina, and A. Parameswaran. Evaluating the crowd with confidence. Technical report, Stanford University, 2012.

[16] D. R. Karger, S. Oh, and D. Shah. Iterative learning for reliable crowdsourcing systems. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *NIPS*, pages 1953–1961, 2011.

[17] L. Keiff. Dialogical logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2011 edition, 2011.

[18] A. Kittur, B. Smus, S. Khamkar, and R. E. Kraut. Crowdforge: crowdsourcing complex work. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 43–52, New York, NY, USA, 2011. ACM.

[19] J. Kleinberg and E. Tardos. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[20] M. Kosinski, Y. Bachrach, G. Kasneci, J. V. Gael, and T. Graepel. Crowd iq: measuring the intelligence of crowdsourcing platforms. In *WebSci'12*, pages 151–160, 2012.

[21] K. Lieberherr. The Scientific Community Game. Website, 2009. `http://www.ccs.neu.edu/home/lieber/evergreen/specker/scg-home.html`.

[22] K. J. Lieberherr and A. Abdelmeged. The Scientific Community Game. In *CCIS Technical Report NU-CCIS-2012-19*, October 2012. `http://www.ccs.neu.edu/home/lieber/papers/SCG-definition/SCG-definition-NU-CCIS-2012.pdf`.

[23] K. J. Lieberherr, A. Abdelmeged, and B. Chadwick. The Specker Challenge Game for Education and Innovation in Constructive Domains. In *Keynote paper at Bionetics 2010, Cambridge, MA, and CCIS Technical Report NU-CCIS-2010-19*, December 2010. `http://www.ccs.neu.edu/home/lieber/evergreen/specker/paper/bionetics-2010.pdf` .

[24] K. J. Lieberherr and E. Specker. Complexity of Partial Satisfaction. *Journal of the ACM*, 28(2):411–421, 1981.

[25] K. J. Lieberherr and E. Specker. Complexity of Partial Satisfaction II. *Elemente der Mathematik*, 67(3):134–150, 2012. `http://www.ccs.neu.edu/home/lieber/p-optimal/partial-sat-II/Partial-SAT2.pdf`.

[26] M. Marion. Why Play Logical Games. Website, 2009. `http://www.philomath.uqam.ca/doc/LogicalGames.pdf`.

[27] P. Minder and A. Bernstein. Crowdlang - first steps towards programmable human computers for general computation. In *Proceedings of the 3rd Human Computation Workshop*, AAAI Workshops, pages 103–108. AAAI Press, 2011.

[28] J. Petit, O. Giménez, and S. Roura. Jutge.org: an educational programming judge. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, SIGCSE '12, pages 445–450, New York, NY, USA, 2012. ACM.

[29] K. R. Popper. *Conjectures and refutations: the growth of scientific knowledge, by Karl R. Popper*. Routledge, London, 1969.

[30] W. Scacchi. The Future of Research in Computer Games and Virtual Worlds: Workshop Report. Technical Report UCI-ISR-12-8, 2012. `http://www.isr.uci.edu/tech_reports/UCI-ISR-12-8.pdf`.

[31] R. Sedgewick. The Role of the Scientific Method in Programming. Website, 2010. `http://www.cs.princeton.edu/~rs/talks/ScienceCS.pdf`.

[32] G. Smith, J. Whitehead, and M. Mateas. Tanagra: a mixed-initiative level design tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 209–216, New York, NY, USA, 2010. ACM.

[33] TopCoder. The TopCoder Community. Website. `http://www.topcoder.com/`.

[34] T. Tulenheimo. Independence friendly logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2011 edition, 2011.

[35] K. J. S. Zollman. The communication structure of epistemic communities. *Philosophy of Science*, 74(5):574–587, 2007.