The Scientific Community Game: A Lens to Focus the Global Brain

Ahmed Abdelmeged
Northeastern University
mohsen@ccs.neu.com

Karl Lieberherr
Northeastern University
lieber@ccs.neu.edu

ABSTRACT

We define the Scientific Community Game (SCG, formerly called the Specker Challenge Game) and we show basic game properties which are key to making the game interesting. An example illustrates the concepts. SCG, the first generic model of the Popperian Scientific Method on the web, models scientific communities and has broad applications in teaching and research organization.

SCG is designed to be both educational for scholars, and to solve problems that we don't know how to solve yet. SCG provides a systematic framework to develop and disseminate the world's constructive claims in formal scientific domains. The development of claims is both collaborative and self-evaluating, and SCG is an effective lens to focus the global brain on solving a specific problem.

A key contribution of the paper is the design of SCG, including a CSP-model of payoff function design. The CSP-model allows us to state and prove key properties of the game across all its instantiations.

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. INTRODUCTION

Popper, one of the most prominent philosophers of science of the previous century, promoted in "Conjectures and Refutations" the idea that each hypothesis should have a description how it could be refuted, and that scientific knowledge grows through the introduction of refutable hypothesis followed by intensive testing of their correctness.

We designed the Scientific Community Game (SCG) to encourage scholars to put as much effort and intelligence as possible into proposing claims that are hard to falsify as well as into attempting to refute those claims. The gamification of science that we propose has several **benefits**: (1) We make scientific activity available to a wider audience. Trying to refute a claim is a simple activity. (2) We get psychological advantages: playing the game is fun and leads to the discovery of what is known within the current group of players. (3) We make it easy to expose claims to refutation attempts, which leads to better science in the long run [24]. (4) The lab definition mechanism of the SCG makes it easy to focus a group of people on a specific problem. (5) Popper's ideas become applicable to claims in any formal science not only to complex scientific theories.

To be submitted for review to the Serious Games track of the conference Foundations of Digital Games, 2013, Crete.

1.1 SCG in a Nutshell

In the SCG, a lab consists of a set of refutable claims [24]. Scholars take on the roles of proponents and opponents of the claims in the lab. Opponents attempt to refute claims against proponents. A refutation attempt initiates an orderly dialog between the proponent and opponent, which induces collaborative behavior. In the SCG, successfully refuting a claim against the claim's proponent does not mean that the claim is false. It only means that the proponent might not have the skills to defend the claim. The labs are self-evaluating in that there is no need of a third party to evaluate the contributions of the scholars: the proponent and opponent evaluate each other fairly.

There are two classes of SCG users, lab designers and scholars (players).

- Lab designers have a problem they want to focus scholars on solving through specializing SCG. The role of lab designer can be played by a researcher who needs a specific problem solved or an educator who wants her students to practice what they were taught. Lab designers define a few sets and functions that define the problem to be solved. As return on their investment they get a lab where scholars are invited to participate to solve the problem. Defining a lab can be viewed as writing a program for the global brain [6].

- Scholars inhabit a lab, either because they are enticed by an educator or because they have good skills to solve the problem defined by the lab. A scholar has the opportunity to influence the quality of the knowledge base and the quality of the lab procedures. Scholars are evaluated in the lab based on the quality of their contributions compared to the contributions of their peers.

  The role of scholar can be played by a human or an avatar (software). The avatar variant works only for labs that are sufficiently well understood.

1.2 SCG Applications

SCG has several **applications**, including:

1. problem solving and research in formal science. Funding agencies, such as NSF, define, in collaboration with interested researchers, labs that define the problem to be solved.

Through playing the game, NSF builds a knowledge base of refutable claims and refutation attempts. Furthermore, the self-evaluating nature of SCG will fairly evaluate the contributions of scholars and the collaborative nature will lead to productive team work. Newcomers can contribute by participating in a long-running lab (dozens of years).

2. teaching (traditional, online and massively open online) courses in STEM areas. To teach a particular problem solving skill, we design a lab for the problem. Playing SCG challenge the students' self-image about their ability to solve the lab's problem. Thus, encouraging students to acquire the desired problem solving skill. The self-evaluating nature of SCG helps lifting much of the evaluation from the teacher and allows stronger students to give precisely targeted feedback to weaker students.

3. software development for computational problems. A computational task is defined by a lab where the role of a scholar is played by an avatar (software). Competitions are held, and the winning avatars will contain the best (within this group of competing avatars) algorithms for the computational task.

1.3 Contributions

The main contribution of this paper is a definition of SCG, a generic game for the Popperian Scientific Method [24]. The SCG definition in this paper presents an abstraction of our publicly available implementation [1].

A second contribution of this paper is an analysis of SCG that shows that it is consistent with its design goals, and it has several desirable properties that make the game interesting.

1.4 Organization

This paper is organized as follows: In section 2 we define binary SCG while in section 3 we introduce SCG competitions that build on binary SCG. In section 4 we describe SCG from the perspective of current game design models. In section 5 we analyze the binary game and show that it has properties that make the game interesting. In section 6, we present our experience with SCG. In section 7, we present some of the related work. In section 8 we present some of our future work. Section 9 concludes the paper.

2. BINARY GAME DEFINITION

We first define the binary SCG which is the building block for competitions. The game is played between two scholars (players), Alice and Bob, disputing the correctness and optimality of claims about a particular domain. We define the binary SCG through its extensive-form (or tree-form) representation which is a common representation in game theory [16]. We start by giving some auxiliary definitions for *Domain*, *Refutation Protocol*, *Lab*, *Claim* and *Refutation Game* that we eventually use for defining the Binary SCG.

2.1 Domain

A *Domain* consists of:

1. a set *Instance* of (problem) instances,

2. a set *Solution* of (problem) solutions,

3. a predicate *valid(i : Instance, s : Solution) : Boolean* that holds when the solution *s* is valid for the instance *i*, and

4. a function *quality(i : Instance, s : Solution) : Real* that returns a real number representing the quality of the solution *s* to the instance *i*.

When designing a domain the *valid* function should be as lax as possible such that valid solutions for any problem can be found with minimal to no intelligence. The reason is that failing to provide a *valid* solution is used as an indicator that a player (which can be an avatar) is out of order and that it should be kicked out of the lab immediately. It should also be kept in mind when designing the *quality* function that the game interprets higher quality as being better.

Examples of domains include, the domain of CNF Satisfiability where *Instance* is the set of CNF formulae, *Solution* is the set of variable assignments to booleans. A solution *s* is valid for an instance *i* if and only if *s* provides a boolean value for all variables mentioned in *i*. A potential definition of *quality(i,s)* is the fraction of clauses in *i* satisfied by *s*.

When there are multiple valid solutions for a given problem, the $quality$ function is used to discriminate between them. For example, for the domain where $Instance$ is the set of sequence alignment and $Solution$ is the set of all sequence alignment algorithms such as Smith-Waterman and BLAST,...etc, we could be concerned with full sequence alignment in which case, Smith-Waterman might have a higher quality. If we are interested in trading some accuracy for speed, BLAST might be of a Higher quality.

2.2 Refutation Protocol

A refutation protocol is a sequence of actions to be taken by two players, a proponent and an opponent of a particular claim, in order to resolve a dispute about whether the claim holds or not. The actions can be either to provide an instance or to provide a solution to a particular instance. Protocols can be defined by the following grammar:

```
ProtocolSpec =  <steps> CommaList(Step).
Step = <actor> Actor <action> Action.
Actor = Proponent | Opponent.
Proponent = "P".
Opponent = "O".
Action = ProvideAction | SolveAction.
ProvideAction = "I" "[" <instanceId> int "]".
SolveAction = "S" "[" <solutionId> int "]"
  "of" <instance> ProvideAction.
```

An example of a protocol is $P : I[0], O : S[1]\ of I[0]$ in which the proponent provides an instance named $I[0]$ and the opponent provides a solution named $S[1]$ for the instance $I[0]$.

2.3 Lab and Claim

Labs play a central role in SCG. SCG games are played in the context of a particular lab. A lab focuses the scientific discourse of games played in it through defining a family of claims about a particular domain. The claims in a lab have the same structure and only differ in the claim parameter values. Labs require a great care to define and effectively resemble a declarative specification of a (human) computation [6] that separates true from false claims in the lab and also finds the optimal claims in optimization labs.

A *Lab* consists of:

1. a $Domain : d$ that gives a context for claims defined in the lab;

2. a refutation protocol $proto : Protocol$, used to resolve disputes regarding the correctness of claims;

3. a claim structure $Claim$ consisting of:

   (a) a number of claim parameters;

(b) an instance set predicate $isp(i : d.Instance) : Boolean$ defining a family of instance sets. Each set in the family is a subset of *d.Instance* and is chosen by claim parameter values. The intuition is that a claim c asserts a property about the set $\forall i \in d.Instance : c.isp(i)$.

(c) a refutation predicate $p(I : d.Instance[], S : d.Solution[]) : Boolean$ defines the property that the claim asserts about the family of sets defined by isp. The protocol together with the refutation predicate are well-formed, if every ProvideAction is used by at least one later SolveAction or the refutation predicate and every SolveAction refers (through the step number) to a ProvideAction. The output of every SolveAction must be used in the refutation predicate.

(d) a predicate $stronger(c_2 : Claim) : Boolean$ defining a partial order on claims in the lab.

(e) a function $distance(c_2 : Claim) : Real$ defining the distance between two claims in the lab provided that the one of the claims is stronger than the other.

Consider a lab with the following protocol instance $P : I[0], O : S[1]$ of $I[0]$. A claim c in this lab intuitively means: "I, the proponent, can give an instance $I[0]$ where $c.isp(I[0])$ holds, such that for any solution *S[1]* of *I[0]* given by you, the opponent, where *valid(I[0],S[1])* holds, the refutation predicate $p(I, S)$ holds.". This claim can also be expressed as a mathematical statement about the underlying domain as: "There exists an instance *I[0]* where $c.isp(I[0])$ holds, such that for all solutions *S[1]* of *I[0]* where *valid(I[0],S[1])* holds, the refutation predicate $p(I, S)$ holds.". In order to support this claim, the proponent has to deliver $I[0]$ and opponent has to deliver *S[1]* in a resource constrained environment.

If we append the following protocol step to the end of the above protocol $P : S[2]$ of $I[0]$ and define the refutation predicate to be: $p(I, S) = (Lab.d.quality(S[2], I[0]) \geq Lab.d.quality(S[1], I[0]))$, we get a very different kind of claim. It says that proponent is at least as good as opponent in solving instances in the given lab. Which is a statement about the performance of players rather than about the underlying domain.

2.3.1 Classification of Claims and Labs

We define a claim to be **true** if it has a support strategy for the proponent. This means that, for true claims, the proponent can always avoid refutations. We define a claim to be **false** if it has a refutation strategy for the opponent, i.e., the opponent can always succeed in refuting. We define a claim to be **indeterminate** if it has neither a defense nor a refutation strategy.

We classify labs into **optimization**, **standard**,and **bivalent** labs. Optimization labs have no restrictions. Standard labs are where claims are not comparable for strength (i.e. the $stronger$ predicate is defined to be $false$). Bivalent labs are a special case of standard labs with no indeterminate claims.

2.4 Refutation Game

To attempt a refutation of a given claim c, both the proponent pro and the opponent opp of the claim enter into a refutation game $refute(c : Claim, pro : Scholar, opp : Scholar)$.

The refutation game consists of an exchange of instances and solutions as defined by the protocol $c.Lab.proto$ where the proponent P and opponent O in the protocol are bound to the scholars pro and opp respectively. The exchanged instances must be in the set $\forall i \in c.d.Instance : c.isp(i)$. A solution $S[n]$ for instance $I[m]$ must be valid for that instance. i.e. $c.d.valid(I[n], S[m])$ must hold. The refutation predicate is used to decide the winner of the refutation game. If the $c.p(I, S)$ holds, the proponent has successfully supported the claim and wins. Otherwise, the opponent has successfully refuted the claim and wins. I, S refer to the instances and solutions exchanged during the refutation game. As a general rule, solutions are all kept secret until protocol evaluation. When a scholar provides a solution she does not know about solutions provided by the other scholar.

2.5 Binary SCG

The first move is performed by the scholar taking the role of the proponent, and it is to propose a claim c from the claims in the lab. By doing so, the proponent asserts that c is a true claim that there is no other claim in the lab that is stronger. In Figure 1, the first move is denoted by a thick dashed blue line to indicate that it summarizes a tree with single edge branches, an edge for each possible claim c.

The second move is performed by the opponent, and it is to decide whether to:

1. dispute the correctness of c, denoted *disputeD(c)*, or to
2. dispute the optimality of c. In this case, the opponent must provide a stronger claim c'. This decision is denoted *strengthenD(c, c')*, or to
3. agree with c, denoted *agreeD(c)*.

Based on the second move, the proponent and the opponent enter into a different refutation game, which is denoted by a thick solid red line to indicate that it summarizes a tree with branches of arbitrary length. This is consistent with our design goal of encouraging refutation attempts. If the second move is *disputeD(c)*, the refutation game is $refute(c$, proponent, opponent). If the second move is *strengthenD(c, c')*, the refutation game is $refute(c'$, opponent, proponent). If the second move is *agreeD(c)*, the refutation game is $refute(c$, opponent, proponent). Even though, there is no dispute, the two players engage in a refutation game to test whether the opponent is indeed able to support c or it is only trying to avoid entering into a dispute with the proponent over c.

Scholars must make their moves within a given resource (time, space, ...etc) limit. The winner of the binary SCG, is the scholar that achieve the highest payoff. Payoff is discussed in section 2.5.1.

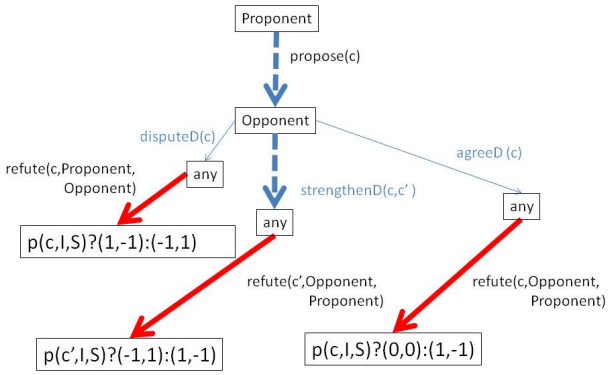

Figure 1: SCG Binary Game Tree.

2.5.1 Payoff function

Payoff is determined by: (1) the decision of the opponent in the second move. (2) the value of a refutation predicate which has as parameters the values collected along the current path back to the root. Including either the proposed claim c or the strengthened claim c', as well as instances I and solutions S provided during the refutation game.

We use the following notation to describe the payoff function: $c.p(I,S)?(\mathrm{Pt},\mathrm{Ot}),(\mathrm{Pf},\mathrm{Of})$ If $c.p(I,S)$ is true, Pt is the payoff for the proponent and Ot is the payoff for the opponent. If $c.p(I,S)$ is false, Pf is the payoff for the proponent and Of is the payoff for the opponent.

1. If the opponent decision is *disputeD(c)*, the payoff is $c.p(I, S)\ ?\ (1,-1) : (-1,1)$. The rationale is that if the predicate holds, the proponent has successfully supported c and gets a point for that. The proponent failed to refute c and loses a point. If the predicate is false, the opponent has failed to support c and loses a point. The opponent has successfully refuted the c and gets a point.

2. If the opponent decision is *strengthenD(c, c')*, the payoff is $c'.p(I, S)\ ?\ (-d, d) : (1, -1)$. where $d = c.distance(c, c')$. The rationale is that if the predicate holds, the opponent gets rewarded because she successfully supported her proposed stronger claim c'. The payoff is proportional to the amount of strengthening as specified by the distance function to encourage more strengthening. If the predicate does not hold, the proponent has successfully refuted the stronger claim and gets rewarded with a point. The strengthening was not successful.

3. If the opponent decision is *agreeD(c)*, the payoff is $c.p(I, S)?\ (0,\ 0) : (1,\ -1)$. The rationale is that if the predicate holds, the opponent has successfully supported the claim and nobody gets a point because no dispute has taken place. Agreement is successful. If the predicate does not hold, the opponent has failed to support the claim and loses a point. Agreement is not successful.

2.6 Example

Fig. 2 (top left) provides a summary of the *Domain* structure. Fig. 2 (top right) provides an example of a *Domain*, the Gale-Shapley worst case domain. In this domain, *Instance* is the set of natural numbers, *Solution* is the set of preferences which is the input to the Gale-Shapley matching algorithm. A solution *s* is valid for an instance *i* if and only if *s* is syntactically well formed and it gives preferences for *i* men and women. The quality of a solution *s* to an instance *i* is the number of iterations of the while loop in the Gale-Shapley matching algorithm when supplied the preferences *s* as input.

Fig. 2 (middle left) provides a summary of the *Lab* structure and provides an example *Lab* (middle right), the Gale-Shapley worst case lab. *Claim*s in this lab have two parameters n and q. n is the number of men and women. q is the number of iterations of the Gale-Shapley matching algorithm that can be achieved by giving a preference for n men and women. The instance set predicate of the claim selects only one instance, namely the instance where the number of men and women is the same as the claim parameter n. The refutation protocol is that the opponent should provide an instance $I[0]$ satisfying the instance set predicate of the claim (there is only a singleton instance) and the proponent should provide a preference $S[1]$ for $I[0]$. The refutation predicate holds if $S[1]$ causes the Gale-Shapley matching algorithm to iterate at least q times. The $stronger$ predicate holds if the current claim has a higher value for the q parameter. The distance function $distance$ is defined as the difference between the q parameter of the current and the given claim. Several other examples in the style of Fig. 2 are in the supplementary materials [2].

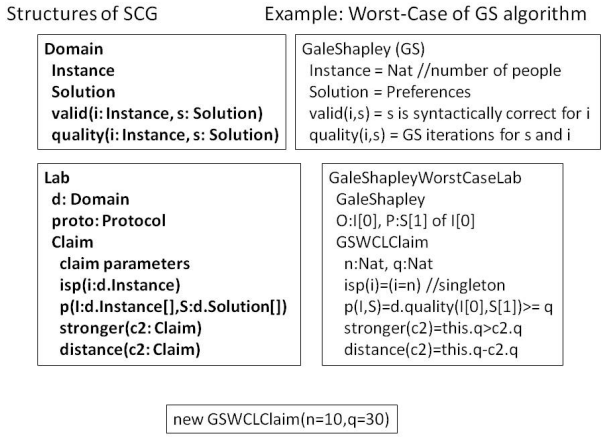

Figure 2: SCG Structure.

3. BEYOND BINARY SCG

3.1 Matches and Tournaments

To alleviate potential first-move-advantage in binary SCG, we propose that players engage in matches rather than binary games. A match consists of a number of rounds of binary SCG where players switch the roles of opponent and proponent. Tournaments (e.g. round-robin, knock-out, Swiss-style) enable more than two players to engage in a single competition. Round-robin tournaments are more suitable for avatars playing SCG because they involve $n.(n-1)$ matches where n is the number of players. Swiss-style tournaments are more suitable for humans playing SCG because it involves fewer games.

3.2 Reputation Score

In real scientific communities, scientists build up their reputation based on their work and the breakthroughs they achieve. Likewise, in SCG, we propose to compute reputation scores, achieved in a particular lab, for scholars based on their performance in competitions held in that lab as well as on their breakthroughs.

3.2.1 Breakthroughs

Being the first in a scientific community is crucial for the reputation of a scholar. Therefore, the SCG as a faithful model of a scientific community, also deals with being the first.

The breakthrough metric is computed after a competition or a series of competitions as a retrospective. For each important event, a time stamp is stored. Important events are: refutation, support and strengthening. We assume that we have the set of claims believed to be true (*BelievedTrue*) or false (*BelievedFalse*) or optimal (*BelievedOptimal*).

For each claim in *BelievedTrue* or *BelievedOptimal*, we find the first scholar who proposed the claim and supported it. That scholar gets one point added to the breakthrough component of their reputation.

For each claim in *BelievedFalse*, we find the first scholar who successfully refuted the claim. That scholar gets one point added to the breakthrough component of their reputation.

For each claim in *BelievedTrue*, but not in *BelievedOptimal*, we find the first scholar who successfully strengthened the claim and supported the strengthened claim. That scholar gets one point added to the breakthrough component of their reputation.

3.3 Histories and Open Publication

By publishing number of times a particular claim gets supported and refuted and the reputations of players supporting or refuting these claims, SCG creates social welfare.

When a lab gets into a stable state (a sub-optimal equilibrium where breakthroughs cease to happen for particular period of time), it is time to publish the current, maybe imperfect refutation and support strategies. If the scholars are avatars, their software is published. If the scholars are humans, an informal description of their techniques is published.

This levels the playing field and sets the stage for the next advancements. It is important to reward the scholars for their previous investment and not force them too early to publish their techniques.

4. DISCUSSION

In this section we show how binary SCG can be seen from the perspective of current game design models.

4.1 Five Element Game Ontology

Using the five element game ontology by Jose Zagal and Michael Mateas et al. [28]: interface, rules, goals, entities, and entity manipulation we give an overview of the SCG. The **entities** manipulated by the scholars are instances in *Instance*, solutions in *Solution* and claims in *Claim* in the context of a *Lab* and the reputation (payoff points) that a scholar accumulates. The **interface** for a scholar consists of the actions: *propose* a claim, decide to *refute*, *strengthen* or *agree* with a claim, to *provide* an instance and to *solve* an instance. The **rules** of the SCG are the rules of the Scientific Method: you must follow the refutation approach defined by *Lab*. You also must follow the scientific discourse prescribed by the extensive form representation of the game. The **goal** of the game is to make the refutation protocol predicate true or false, depending on the role you play (proponent or opponent of a claim). The **entity manipulation** includes solving instances with a certain quality and creating claims and instances. Gameplay is segmented into binary games and competitions and into increasingly more complex claims which become harder and harder to defend.

4.2 Triadic Game Design

Using the model of Triadic Game Design [11] by Casper Harteveld, we break down the SCG into a Reality, Meaning and Play component. Triadic Game Design describes an approach to serious game design and the SCG defines a large family of serious games. The **Reality** component of the SCG models a scientific community based on the Popper-style Scientific Method [24]. The **Meaning** component consists of cleaning the knowledge base of false or non-optimal claims by developing new constructions which are better than the constructions of others. The **Play** component of the SCG consists of competition, collaboration and finding a treasure (a new construction).

4.3 Exploratory-Performatory Games

According to Jonas Linderoth [21] games challenge two aspects of human nature: our ability to choose appropriate actions and our ability to perform appropriate actions. [21] views gaming as a cycle between interrelated exploratory and performatory actions.

What are the exploratory and performatory actions in the SCG? **Exploratory** actions are: (1) proposing a claim which means choosing from a set of claims. (2) Choosing an action: dispute, agree or strengthen a given claim. **Performatory** actions are: (1) the proponent should defend the proposed claim. (2) If an opponent decides to refute, he should succeed in refuting, etc. In the SCG we also have a cycle of interrelated exploratory and performatory actions. Indeed, this cycle is the dominant activity in the SCG.

5. BINARY SCG ANALYSIS

In this section, we characterize blameable moves in the binary SCG game tree. We argue that blame is both fair and is consistent with the design goal of encouraging scholars to put as much effort and intelligence as possible into proposing claims that are hard to falsify as well as into attempting to refute those claims. We show that the payoff is sound, fair, and competitive with respect to the blame assignment. These three properties are important for SCG to be interesting. For space reasons, we deal in this subsection only with bivalent labs.

5.1 Blame Assignment

First, we divide the branches of the SCG game tree into 8 different sets based on 3 properties. Then, we assign blame to these sets. Figure 3 presents these sets. The three properties are: (1) whether the proponent proposed a true claim or not. This is represented in Figure 3 by the column **claim**. A value of `T` means the proposed claim is true. A value of `F` means the proposed claim was false. (2) the decision made by the opponent. This is represented in Figure 3 by the column **dec**. A value of `a` means agree and a value of `d` means dispute. There is no strengthening in bivalent labs. (3) the outcome of the refutation game and the winner. This is represented in Figure 3 by the column **out**. The outcome is either `s` for "support" or `r` for "refutation" and the winner is either `P` for the "proponent" or `O` for the "opponent".

Figure 3 has two blame columns **cB**, and **oB** with values that are either `P` for the "proponent" or `O` for the "opponent". In the **cB** column, we blame the proponent for proposing a false claim (rows 1, 3, 5, 7). This is consistent with our design goal of encouraging scholars to put effort and intelligence into proposing claims that are hard to falsify. This blame is fair because it is in the hands of the proponent to avoid. In the **oB** column, we blame the player that has lost the refutation game when it is fair to do so. This is consistent with our design goal of scholars to put effort and intelligence into attempting to refute claims. The blame in this column is fair because it is in the hands of both players to avoid. The opponent gets blamed for failing to refute a claim it disputed (rows 3, 4) and for failing to support a claim it agreed with (rows 5, 6). The proponent gets blamed for failing to support a claim it proposed (rows 7, 8). The proponent does not get blamed for failing to refute a claim it proposed (rows 1, 2).

5.2 Payoff Function

In the design of the SCG payoff function there are three forces at work that need to be balanced. We would like the payoff function to be fair, sound and competitive.

5.2.1 Incomplete Knowledge and Fairness

Unfortunately, it is not always decidable whether a claim is true or false. Therefore, the payoff has to be the same for branches (rows of Figure 3) that only differ in the **claim** property (namely, rows {1, 2}, {3, 4}, {5, 6} and {7, 8}). For the payoff to be fair, it

has to give scholars the benefit of the doubt (i.e. the payoff function has to only penalize a scholar for a group of rows where the scholar is blamed in every row of the group). For example, it is unfair to blame either the proponent or the opponent for rows $\{1, 2\}$ because none of the players are blamed in row 2. The payoff function shown in Figure 3 is fair.

5.2.2 *Soundness*

For the payoff to be sound, there must be a chance that a scholar gets eventually penalized if it makes a blameable move. For example, suppose that the proponent proposed a false claim. There are 4 possible branches that start with the proponent proposing a false claim, and they correspond to rows 1, 3, 5, 7 in Figure 3. To be sound, the payoff function must penalize the proponent in one of these rows. The payoff function shown in Figure 3 penalizes the proponent in row 7. The payoff function shown in Figure 3 is sound.

5.2.3 *Competitiveness*

For the payoff to be competitive, it must encourage competition until the last move (i.e. the last move must matter to the payoff). In our case, the last move consists of playing the refutation game. The payoff function must encourage scholars to win the refutation game. For example, the payoff function should favor the proponent in row 5 more than it does in row 1. Because, both rows only differ in the refutation game winner which is the proponent for row 5 and the opponent for row 1. The payoff function shown in Figure 3 is competitive.

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

5.3 Payoff Function Design using CSP

To study the space of payoff functions, we introduce a constraint satisfaction problem that expresses the constraints implied by the fairness, soundness and competitiveness properties. Figure 3 introduces eight variables for formulating the constraints for the payoff function: $paso$, $oaso$, etc. The variable names come from the decision and outcome column.

To penalize the proponent we add the constraint that its payoff must be negative. We say that the difference between the proponent and opponent payoffs is the amount that the payoff function favors the proponent with.

The constraints for the fairness property are: $paso \geq 0$, and $oaso \geq 0$, and $pdsp \geq 0$, and $parp \geq 0$, and $odro \geq 0$..

The constraints for soundness we give separately for oB-soundness (soundness based on blame in the oB column) and cB-soundness (soundness based on blame in the cB column). The constraints for the oB-soundness property are: $odsp < 0$, and $oarp < 0$, and $pdro < 0$..

For cB-soundness, there is one constraint which is: $((pdro < 0) \vee (parp < 0) \vee (pdsp < 0) \vee (paso < 0))$.

The constraints for competitiveness are: $parp - oarp > paso - oaso$ and $pdsp - odsp > pdro - odro$.

Any variations of the payoff function must satisfy all constraints introduced above for the game to be fair, sound and competitive. The above constraints are also useful when only a subset of the properties is desired.

6. 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.

6.1 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 [15] where the best avatars needed to solve a constrained search problem using a modified Pascal triangle.

6.2 Algorithms

The most successful course (using [15] 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.

7. RELATED WORK

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

7.1 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 [5].

Scientific discovery games, such as FoldIt and EteRNA, are variants of the SCG. [7] describes the challenges behind developing scientific discovery games. [4] 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.

7.2 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 [22], [12] have a long history going back to Socrates. The SCG is an imperfect information game which builds on Paul Lorenzen's dialogical games [14].

7.3 Foundations of Digital Games

A functioning game should be deep, fair and interesting which requires careful and time-consuming balancing. [13] 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.

7.4 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 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 [25] 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).

7.5 Online Judges

An online judge is an online system to test programs in programming contests. A recent entry is [23] 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.

7.6 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 instances. Another way is to pair the learner with another learner who is stronger. [3] 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.

7.7 Formal Sciences and Karl Popper

James Franklin points out in [10] 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 [24] is the father of non-deductive methods in science. The SCG is a way of doing science on the web according to Karl Popper.

7.8 Scientific Method in CS

Peter Denning defines CS as the science of information processes and their interactions with the world [8]. The SCG makes the scientific method easily accessible by expressing the hypotheses as claims. Robert Sedgewick in [26] 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.

7.9 Games and Learning

Kevin Zollman studies the proper arrangement of communities of learners in his dissertation on network epistemology [29]. 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.

7.10 CSP-based Game Design

CSP is increasingly being used in the procedural content generation (PCG) community, although not in industry. For example, Tanagra[27] uses a numerical constraint solver to guarantee level playability. In addition, Magy El-Nasr used constraint solving for lighting and adaptive systems for games [9].

7.11 Origins of SCG

A preliminary definition of the SCG was given in a keynote paper [18]. [17] gives further information on the Scientific Community Game. The original motivation for the SCG came from the two papers with Ernst Specker: [19] and the follow-on paper [20]. 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.

8. 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.

9. 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.

10. 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] A. Abdelmeged and K. J. Lieberherr. The scientific community game: supplementary materials. Website, 2012. http://www.ccs.neu.edu/home/lieber/papers/SCG-definition/supplementary/ .

[3] 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.

[4] 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.

[5] 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.

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

[7] 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.

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

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

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

[11] C. Harteveld. *Triadic Game Design*. Springer, 2011.

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

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

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

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

[16] K. Leyton-Brown and Y. Shoham. Essentials of game theory: A concise multidisciplinary introduction. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2(1):1–88, 2008.

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

[18] 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 .

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

[20] 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.

[21] J. Linderoth. Why gamers don't learn more: An ecological approach to games as learning environments. In L. Petri, T. A. Mette, V. Harko, and W. Annika, editors, *Proceedings of DiGRA Nordic 2010: Experiencing Games: Games, Play, and Players*, Stockholm, January 2010. University of Stockholm.

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

[23] 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.

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

[25] 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.

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

[27] 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.

[28] J. P. Zagal, M. Mateas, C. Fernandez-Vara, B. Hochhalter, and N. Lichti. Towards an ontological language for game analysis. In *in Proceedings of International DiGRA Conference*, pages 3–14, 2005.

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