ECAI 2014
T. Schaub et al. (Eds.)
© 2014 The Authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-419-0-225

On the Efficient Implementation of Social Abstract Argumentation

Marco Correia and **Jorge Cruz** and **João Leite** [1]

Abstract. In this paper we present a novel iterative algorithm – the Iterative Successive Substitution (ISS) – to efficiently approximate the models of debates structured according to *Social Abstract Argumentation* [10]. Classical iterative algorithms such as the Iterative Newton-Raphson (INR) and the Iterative Fixed-point (IFP) don't always converge and, when they do, usually take too long to be effective. We analytically prove convergence of ISS, and empirically show that, even when INR and IFP converge, ISS always outperforms them, often by several orders of magnitude. The ISS is able to approximate the models of *complex* debates with thousands of arguments in well under a second, often in under one tenth of a second, making it comfortably suitable for its purpose. Additionally, we present a small modification to ISS that, with a negligible overhead, takes advantage of the topological structure of certain debates to significantly increase convergence times.

1 Introduction

Social Abstract Argumentation [10] extends Dung's Abstract Argumentation with votes, and was introduced to help structure online debates and foment wide participation with different levels of engagement: while some more eager participants – experts or enthusiasts – would specify arguments and their attack relations, others, who simply want to follow the debate, but who nevertheless have an opinion on what is happening and wish to express it and influence the outcome, could do it by voting on arguments (and, according to some extensions [6], also on attacks).

According to Social Abstract Argumentation, arguments and attack relations that form a debate are arranged in a directed graph, with each argument having associated a number of positive and negative votes. The semantics of Social Abstract Argumentation assigns one or more models to debates – each of which indicating the *social strength* of each argument, taking into account both the structure of the argumentation graph and the votes – and is parameterized by a set of operators that characterize how votes should be interpreted, the effect of attacks, and how multiple attacks should be combined.

Whereas the semantics of Social Abstract Argumentation was defined at a very general level and is, in fact, a family of semantics, each specified by a specific set of operators, one particular instance was studied in greater detail. It is specified by operators based on the product T-norm and its dual, the probabilistic sum T-conorm, and enjoys many desirable properties such as the fact that argument *social strength* is gradual, going beyond the classical true/accepted or false/defeated and is limited by popular opinion where every vote counts. Of key importance is the fact that, with this semantics, every debate has a model, and it is strongly believed – though a proof is still missing (see conjecture 14 in [10]) – that this model is unique.

This semantics and equivalent/restricted variants have been investigated in [10, 6, 4, 7, 2]. However, an online debating system based on Social Abstract Argumentation is yet to be deployed, and very little has been investigated with respect to its implementation and computational properties. One possible reason may be the complexity of solving the highly non-linear system of equations that characterizes the semantics. [10] discusses the possibility to use a classical iterative fixed-point algorithm, but only for a very restricted class of debates where, otherwise, it would not converge to the solution, thus not usable in general. Other classical algorithms used to solve systems of non-linear equations run into difficulties because a) they often converge to a solution outside the domain of the problem (e.g. assigning negative *social strengths* to arguments, instead of values drawn from $[0, 1]$) and b) are very inefficient due e.g. to the need to determine the Jacobian of matrices, which may be prohibitive for large systems. One such algorithms is the iterative Newton-Raphson which often takes more than 5 min to approximate the solution of a debate with 5000 arguments and a reasonable number of attacks between them, thus not usable in a wider scale.

Despite the inexistence of such wide scale online debates of the kind envisioned by [10] to allow us to assess their size, our goal is to be able to approximate the solution of a debate with 5000 arguments and a reasonable number of attacks between them in under 1 sec.

In this paper, we present a novel iterative algorithm, dubbed *Iterative Successive Substitution*, and investigate its properties, of which we highlight:

- convergence to the (believed unique) solution to the debate;
- high efficiency, outperforming the other algorithms when they converge, often by several orders of magnitude, and ability to approximate the models of *complex* debates with thousands of arguments in well under a second, often in under one tenth of a second;
- ability to use the topological structure of the graph, with minimal overhead, to significantly increase convergence times.

The remainder of the paper is structured as follows. In Section 2 we briefly review Social Abstract Argumentation. In Section 3 we review the classical Iterative Fixed-point and Newton-Raphson algorithms. In Section 4, we present the *Iterative Successive Substitution* algorithm, prove it convergence and discuss its rate of convergence. In Section 5 we compare our algorithm with the other two, and investigate its performance on a large number of debates with varying size and complexity. In Section 6 we show how we can take advantage of the topological structure of the debate with the *Iterative Successive Substitution* algorithm. In Section 7 we discuss the relevance and significance of the algorithm beyond [10] and conclude.

[1] CENTRIA & Departamento de Informática, Faculdade Ciências e Tecnologia, Universidade Nova de Lisboa, email: jleite@fct.unl.pt

2 Social Abstract Argumentation

In this section we recap Social Abstract Argumentation Framework [10]. It is an extension of Dung's AAF, composed of arguments and an attack relation to which we add an assignment of votes to each argument. In [6] this is extended to also allow votes on attacks. Here we keep to the original version, but it is import an to stress that the resulting system of equations that needs to be solved to determine the semantics in [6] is of the same form as the one we present here, so the algorithm also applies there.

Definition 1 (Social Abstract Argumentation Framework) *A* Social Abstract Argumentation Framework (SAF) *is a triple $\langle \mathcal{A}, \mathcal{R}, V \rangle$ where $\mathcal{A}$ is a set of arguments, $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is a binary attack relation between arguments and $V : \mathcal{A} \to \mathbb{N} \times \mathbb{N}$ is a total function mapping each argument to its number of positive and negative votes.*

Notation 2 *Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a social abstract argumentation framework. We use $\mathcal{A}_F$, $\mathcal{R}_F$, and V_F to denote, respectively, the set of arguments ($\mathcal{A}$), the attack relation ($\mathcal{R}$), and the votes (V) of F, n_F to denote the number of arguments in $\mathcal{A}_F$, e_F the number of attacks in $\mathcal{R}_F$, and d_F the* density *of the argumentation graph, defined as the probability that $(x_1, x_2) \in \mathcal{R}_F$ for two (distinct) nodes x_1, x_2, selected at random, i.e $d_F = e_F / n_F\,(n_F - 1)$. $\mathcal{R}^-_F(x) \triangleq \{x_i : (x_i, x) \in \mathcal{R}_F\}$ denotes the set of (direct) attackers of argument x. Whenever unambiguous, we drop the subscript F from $\mathcal{A}_F$, $\mathcal{R}_F$, V_F, V^+_F, V^-_F, $\mathcal{R}^-_F$, n_F, e_F and d_F.*

The semantics is parameterised on a number of operators, encapsulated in the following notion of semantic framework.

Definition 3 (Semantic Framework) *A social abstract argumentation semantic framework is a 5-tuple $\langle L, \tau, \curlywedge, \curlyvee, \neg \rangle$ where:*

- *L is a totally ordered set with top element $\top$ and bottom element $\perp$, containing all possible valuations of an argument.*
- *$\tau : \mathbb{N} \times \mathbb{N} \to L$ is a vote aggregation function which produces a valuation of an argument based on its votes.*
- *$\curlywedge : L \times L \to L$ is a binary algebraic operation on argument valuations used to determine the valuation of an argument based on its valuation given by the votes and how weak its attackers are;*
- *$\curlyvee : L \times L \to L$ is a binary algebraic operation on argument valuations used to determine the valuation of a combined attack;*
- *$\neg : L \to L$ is a unary algebraic operation on argument valuations used to determine how weak an attack is.*

We are now ready to define the semantics of SAFs.

Definition 4 (Social Model) *Let F be a social abstract argumentation framework and $\mathcal{S} = \langle L, \tau, \curlywedge, \curlyvee, \neg \rangle$ a semantic framework. A total mapping $M : \mathcal{A} \to L$ is a* social model of F under semantics $\mathcal{S}$, *or* $\mathcal{S}$-model of F, *if*

$$M(x) = \tau(V(x)) \curlywedge \neg \bigcurlyvee \{M(x_i) : x_i \in \mathcal{R}^-(x)\} \qquad \forall x \in \mathcal{A}$$

where $\bigcurlyvee\{x_1, x_2, ..., x_n\} \triangleq ((x_1 \curlyvee x_2) \curlyvee ... \curlyvee x_n)$. We refer to $M(x)$ as the social strength, or value, of x in M, dropping the reference to M whenever unambiguous.

As mentioned before, one particular semantic framework received greater attention because of its properties. It uses a simple vote aggregation function and is based on the well known product T-norm and probabilistic sum T-conorm. It was dubbed Simple Product Semantics in [10] as is defined as follows:

Definition 5 (Simple Product Semantics) *A simple product semantic framework is $\mathcal{S}^{\cdot}_{\varepsilon} = \langle [0,1], \tau_\varepsilon, \curlywedge, \curlyvee, \neg \rangle$ where 1) $x_1 \curlywedge x_2 = x_1 \cdot x_2$, 2) $x_1 \curlyvee x_2 = x_1 + x_2 - x_1 \cdot x_2$, 3) $\neg x_1 = 1 - x_1$ and 4) for some $\varepsilon > 0$, $\tau_\varepsilon(v^+, v^-) = \frac{v^+}{v^+ + v^- + \varepsilon}$*

The meaning of ϵ is explained in [10] and, in practice, it should be a sufficiently small value with no significant influence in result of the voting aggregation function. Throughout the remainder of the paper, $\mathcal{S}$ will always stand for the Simple Product Semantics.

Theorem 6 *[10] Every social abstract argumentation framework F has at least one $\mathcal{S}$-model.*

Conjecture 7 *[10] Every social abstract argumentation framework F has at most one $\mathcal{S}$-model.*

Proposition 8 *Let F be a social abstract argumentation framework, $x \in \mathcal{A}_F$ such that $\tau(x) = 0$, and F' be obtained from F by removing x and all attacks involving x. Let M be an $\mathcal{S}$-model F. Then, $M(x) = 0$. Additionally, M', obtained from M by restricting it to $\mathcal{A}_F \backslash \{x\}$, is a $\mathcal{S}$-model F'.*

The relevance of the previous proposition is that we can ignore arguments without positive votes.

The problem of finding a model according to the simple product semantics can then be cast to the problem of finding a solution of a nonlinear system where variables represent the arguments and equations encode their attacks, with the following generic form:

Definition 9 *A Social Abstract Argumentation System is a square nonlinear system with n variables $\{x_1, \ldots, x_n\}$ and n equations:*

$$x_i = \tau_i \prod_{j \in A_i} (1 - x_j) \qquad 1 \le i \le n \tag{1}$$

where $\tau_i \in\ ]0,1[$ and $A_i \subseteq \{1, \ldots, n\}$.

3 Iterative Algorithms

Contrary to the linear case, systems of nonlinear equations cannot be solved exactly using a finite number of elementary operations. Instead, iterative algorithms are usually used to generate a sequence $(\mathbf{x}^{(k)})_{k \in N_0}$ of approximate solutions. These algorithms start with an initial guess $\mathbf{x}^{(0)}$ and, to generate the approximating sequence, follow an iteration scheme of the form $\mathbf{x}^{(k+1)} = \mathbf{g}(\mathbf{x}^{(k)})$ where the fixed-points for $\mathbf{g}$ are solutions $\mathbf{x}^*$ of the nonlinear system.

The success of iterative algorithms depend on their convergence properties. Given a domain of interest, an iterative method that converges for any arbitrary initial guess is called globally convergent. If convergence is only guaranteed when the initial approximation is already close enough to the solution, the algorithm is called locally convergent. In the case of Social Abstract Argumentation Systems the domain of interest is $]0,1[^n$ thus the iterative algorithm must converge to a solution $\mathbf{x}^* = (x^*_1, \ldots, x^*_n) \in\ ]0,1[^n$.

Even for converging iterations the approximating sequence will never arrive at the exact solution after finitely many steps. Hence an appropriate termination criterion must be defined to stop iterating if a small enough approximation error is predicted. A commonly used criterion is to stop iterating when the changes made by an iteration are below some predefined tolerance:

$$\|x^{(k+1)} - x^{(k)}\|_\infty = \max_{1 \le i \le n} |x^{(k+1)}_i - x^{(k)}_i| < tol \tag{2}$$

A comprehensive treatment of methods for solving nonlinear systems of equations with some recent developments on iterative methods can be found in [12, 1]. Next we present two locally convergent iterative algorithms that can be used to solve Social Abstract Argumentation Systems. In sect. 4 we propose a globally convergent algorithm that outperforms both classical approaches.

In the classical Iterative Fixed-Point Algorithm (IFP) the iteration scheme is directly obtained from the equations (1):

Algorithm 10 (IFP) *The IFP algorithm uses the iteration rule:*

$$x_i^{(k+1)} = \tau_i \prod_{j\in A_i} \left(1 - x_j^{(k)}\right) \tag{3}$$

The IFP algorithm clearly satisfies the requirement that the fixed-points for the iteration rule (3) are solutions of the nonlinear system (1). However the algorithm is only locally convergent and often diverge, even for systems with a reduced number of variables.

The Newton-Raphson method is a classical approach to solve square nonlinear systems of the form $\mathbf{f}(\mathbf{x}) = 0$ which requires the computation of $\mathbf{J}(\mathbf{x})$, the Jacobian matrix of $\mathbf{f}$: $\{\mathbf{J}(\mathbf{x})\}_{ij} = \frac{\partial f_i(\mathbf{x})}{\partial x_j}$.

The application of the Iterative Newton-Raphson Algorithm (INR) to Social Abstract Argumentation Systems imply the reformulation of the equations (1) into the required form:

Algorithm 11 (INR) *The INR algorithm uses the iteration rule:*

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{J}(\mathbf{x}^{(k)})^{-1}\mathbf{f}(\mathbf{x}^{(k)}) \tag{4}$$

with $\mathbf{f}(\mathbf{x}) = (x_1 - \tau_1 \prod_{j\in A_1} (1-x_j), \ldots, x_n - \tau_n \prod_{j\in A_n} (1-x_j))^t$.

The INR algorithm is expected to converge fast (quadratically), provided that a sufficiently accurate initial guess is known and the inverse of the Jacobian matrix exists. However the computation of the Jacobian at each iteration may be prohibitive for large systems.

4 Iterative Successive Substitution Algorithm

To solve Social Abstract Argumentation Systems we propose the Iterative Successive Substitutions Algorithm (ISS) which is an adaptation of the Gauss-Seidel method for systems of nonlinear equations.

Algorithm 12 (ISS) *The ISS algorithm uses the iteration rule:*

$$x_i^{(k+1)} = \tau_i \prod_{j<i, j\in A_i} \left(1 - x_j^{(k+1)}\right) \prod_{j\geq i, j\in A_i} \left(1 - x_j^{(k)}\right) \tag{5}$$

From the initial guess $\mathbf{x}^{(0)}$, elements of $\mathbf{x}^{(k+1)}$ are computed sequentially using forward substitution until the stoping criterion is attained.

4.1 Convergence

In the following we will prove that ISS will always converges to a solution $\mathbf{x}^* \in \ ]0,1[^n$ of system (1), independently of the initial guess $\mathbf{x}^{(0)} \in \ ]0,1[^n$. Assuming the uniqueness conjecture, it will always converge to the unique solution of the system in $]0,1[^n$.

Proposition 13 *If variable x_m converges to x_m^* then system (1) converges to the equivalent system:*

$$x_i = \begin{cases} x_m^*, & i = m \\ \tau_i' \prod_{j\in A_i'} (1-x_j), & i \neq m \end{cases}$$

with: $A_i' = A_i \backslash \{m\}$; $\tau_i' = \tau_i(1-x_m^*)$ *if* $m \in A_i$ *or* τ_i *otherwise.*

Proof. The new system is obtained by rewriting the right hand side of all the variables x_i attacked by x_m updating both τ_i and A_i. ■

Proposition 14 *If there is always at least one variable x_m converging to x_m^* then ISS converges to a solution $\mathbf{x}^* \in]0,1[^n$.*

Proof. According to prop. 13 if there is a variable converging, the system converges to a system composed of: 1) a trivial equation where this variable is fixed and; 2) a subsystem with one less variable and one less equation. Omitting the trivial equations and starting from the original n-system (with n variables and n equations) the existence of a variable converging would make it converge to a $(n-1)$-system. Again, the existence of a variable converging within this $(n-1)$-system would make it converge to a $(n-2)$-system. In the limit, this would guarantee the convergence of the original system to a system of n trivial equations of the form $x_i = x_i^*$. ■

In the following we prove that along the iterations of ISS there is always at least one variable converging and so, according to prop. 14, the algorithm converges to a solution.

Proposition 15 *If $\exists_{1\leq i\leq n} A_i = \emptyset$ then variable x_i converges to τ_i.*

If there are no variables in the conditions of prop. 15 then there must be a sequence of attacks between a subset of system variables that define a cycle.

Definition 16 *System (1) has a cycle characterized by the sequence of m variables $\langle x_{c_1}, \ldots, x_{c_m}\rangle$ if $c_m \in A_{c_1} \wedge \forall_{1\leq i<m} c_i \in A_{c_{i+1}}$.*

Proposition 17 *If $\forall_{1\leq i\leq n} A_i \neq \emptyset$ then system (1) must have a cycle.*

Proof. Starting with $i = 1$ we can choose a member of A_i (it is not empty) and repeat the process until we obtain an element already visited (this is guaranteed because the total number of elements is finite and the process can be indefinitely repeated). ■

Proposition 18 *If system (1) has a cycle characterized by a sequence of m variables $\langle x_{c_1}, \ldots, x_{c_m}\rangle$ then all m variables converge.*

Proof. In the following we assume that there is a cycle characterized by the sequence of m variables $\langle x_{c_1}, \ldots, x_{c_m}\rangle$ and use function $p(c_i)$ to denote the index of the variable that attacks variable x_{c_i} in the cycle: $p(c_i) = c_m$ if $i = 1$ or c_{i-1} otherwise.

Let $A_{c_i}' = A_{c_i} \backslash \{p(c_i)\}$ denote the set of all indexes of variables that attack variable x_{c_i} except the one in the cycle. Let $P_{c_i}^{(k+1)}$ denote the contributions of those attacks in the product of iteration rule (5):

$$P_{c_i}^{(k+1)} = \prod_{j\in A_{c_i}'}^{j<c_i} \left(1 - x_j^{(k+1)}\right) \prod_{j\in A_{c_i}'}^{j\geq c_i} \left(1 - x_j^{(k)}\right)$$

If there is a cycle $\langle x_{c_1}, \ldots, x_{c_m}\rangle$ then for all $1 \leq i \leq m$:

$$x_{c_i}^{(k+1)} = \begin{cases} \tau_{c_i}\left(1 - x_{p(c_i)}^{(k)}\right) P_{c_i}^{(k+1)}, & p(c_i) > c_i \\ \tau_{c_i}\left(1 - x_{p(c_i)}^{(k+1)}\right) P_{c_i}^{(k+1)}, & p(c_i) < c_i \end{cases}$$

Let $l_i = 1$ if $p(c_i) > c_i$ and 0 otherwise. The above simplifies to:

$$x_{c_i}^{(k+l_i)} = \tau_{c_i} P_{c_i}^{(k+l_i)} - \tau_{c_i} P_{c_i}^{(k+l_i)} x_{p(c_i)}^{(k)}$$

Making $l_1^j = \sum_{i=1}^{j} l_i$ and $l = l_1^m$ and rewriting all the equations from $i = 1$ up to m we obtain the generic formula:

$$\begin{aligned} x_{c_m}^{(k+l)} &= \left(\sum_{i=1}^{m} (-1)^{i+1} \prod_{j=m-i+1}^{m} \tau_{c_j} P_{c_j}^{(k+l_1^j)}\right) \\ &+ \left((-1)^m \prod_{j=1}^{m} \tau_{c_j} P_{c_j}^{(k+l_1^j)}\right) x_{c_m}^{(k)} \end{aligned} \tag{6}$$

To illustrate, consider the case where $m = 2$. After another l iterations:

$$\begin{aligned} x_{c_2}^{(k+2l)} &= \tau_{c_2} P_{c_2}^{\left(k+l+l_1^2\right)} - \tau_{c_1} P_{c_1}^{\left(k+l+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l+l_1^2\right)} \\ &+ \tau_{c_1} P_{c_1}^{\left(k+l+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l+l_1^2\right)} \tau_{c_2} P_{c_2}^{\left(k+l_1^2\right)} \\ &- \tau_{c_1} P_{c_1}^{\left(k+l+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l+l_1^2\right)} \tau_{c_1} P_{c_1}^{\left(k+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l_1^2\right)} \\ + \quad & \tau_{c_1} P_{c_1}^{\left(k+l+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l+l_1^2\right)} \tau_{c_1} P_{c_1}^{\left(k+l_1^1\right)} \tau_{c_2} P_{c_2}^{\left(k+l_1^2\right)} x_{c_2}^{(k)} \end{aligned}$$

Notice that after every l iterations the product of the last term will contain two additional factors (of the form τP) and the remaining is a sum of alternate terms with two additional terms. When the number of iterations goes to infinity the last term converges to zero (all factors are positive and smaller than one) and the other terms form an alternating series that is guaranteed to converge accordingly to the Leibniz criterion (it decreases monotonically and goes to 0 in the limit). Since we made no assumptions about the initial values of the variables, variable x_{c_2} is necessarily converging.

In the simplest case where the only variables are x_1 and x_2 and the cycle sequence is $\langle x_1, x_2 \rangle$ with no attacks except those in the cycle: $x_1 = \tau_1 (1 - x_2)$, $x_2 = \tau_2 (1 - x_1)$. In this case all P's are 1, $\tau_{c_1} = \tau_1$, $\tau_{c_2} = \tau_2$, $l_1 = 1$, $l_2 = 0$, $l = 1$:

$$\begin{aligned} x_2^{(k+1)} &= \tau_2 - \tau_1\tau_2 + \tau_1\tau_2 x_2^{(k)} \\ x_2^{(k+2)} &= \tau_2 - \tau_1\tau_2 + \tau_1\tau_2\tau_2 - \tau_1\tau_2\tau_1\tau_2 + \tau_1\tau_2\tau_1\tau_2 x_2^{(k)} \\ \lim_{n\to\infty} x_2^{(n)} &= \lim_{n\to\infty} \left(\sum_{k=0}^{n} \tau_2 (\tau_1\tau_2)^k - \sum_{k=0}^{n} \tau_1\tau_2 (\tau_1\tau_2)^k \right) \\ &+ \lim_{n\to\infty} \left(\prod_{k=0}^{n} (\tau_1\tau_2)^k \right) x_2^{(0)} \\ &= \frac{\tau_2}{1-\tau_1\tau_2} - \frac{\tau_1\tau_2}{1-\tau_1\tau_2} + 0 \times x_2^{(0)} = \frac{\tau_2 - \tau_1\tau_2}{1-\tau_1\tau_2} \end{aligned}$$

The obtained limit is exactly the solution in order to x_2 of the system. Moreover, for any cycle sequence with two variables, a similar proof for the convergence of x_{c_1} can be made by advancing one variable in the cycle and considering the sequence $\langle x_{c_2}, x_{c_1} \rangle$.

In the general case of a cycle sequence with m variables (6), after every l iterations the product of the last term will contain m additional factors (of the form τP) and the remaining is a sum of alternate terms with m additional terms. Similarly to the two variables case, the convergence properties are guaranteed: the last product converge to zero and the alternating series converge to the solution of x_{c_m} independently of the initial guess. Again, by advancing one by one the variables in the sequence, similar proofs can be made for the convergence of any variable in the cycle. ∎

Theorem 19 (Global Convergence) *ISS will always converge to a solution of system (1) independently of the initial guess* $\mathbf{x}^{(0)} \in]0,1[^n$

Proof. Either $\exists_{1\leq i\leq n} A_i = \emptyset$ or $\forall_{1\leq i\leq n} A_i \neq \emptyset$ are true. In the first case, variable x_i converges (prop. 15). In the second case, prop. 17 guarantees that system (1) must have at least one cycle whose variables, accordingly to prop. 18, must converge. In either case there is always some variable converging which, by prop. 14, is sufficient to guarantee the algorithm convergence. ∎

4.2 Convergence rate and time complexity

In this section, we provide an informal study of the convergence rate of ISS. We will analyze the convergence rate with respect to each variable that occurs in a cycle. Variables not occurring in any cycle converge trivially in one iteration after the convergence of all the variables that directly attack them (see prop. 13).

A formal definition for the convergence rate of a sequence of real numbers can be stated as follows.

Definition 20 *Consider a sequence of reals $x^{(k)}$ that converge to a point x^*: $\lim_{n\to\infty} x^{(k)} = x^*$ If positive constants λ and α exist with:*

$$\lim_{n\to\infty} \frac{|x^{(n+1)} - x^*|}{|x^{(n)} - x^*|^\alpha} = \lambda$$

then the order of convergence is α with asymptotic error constant λ.

In the case where $\alpha = 1$ and $\lambda < 1$ the convergence is said to be linear and λ is called the rate of convergence.

When the iterative rule $x^{(n+1)} = g(x^{(n)})$ is continuous and differentiable and converges to point x^*, the computation of its derivative at the convergence point is a classical procedure to verify linear convergency and assess its rate of convergence: If $g'(x^*) \neq 0$ then there is linear convergence and the rate of convergence is $\lambda = |g'(x^*)|$.

The above properties can be used to address the convergence rate of a variable that occurs in a cycle and whose sequence of values are obtained according to the iteration rule (6).

In particular this rate can be computed for the two variables simplest case with no attacks outside the cycle: $x_2^{(n+1)} = g(x_2^{(n)}) = \tau_2 - \tau_1\tau_2 + \tau_1\tau_2 x_2^{(n)}$ and $g'(x^*) = \tau_1\tau_2$ proving linear convergence ($\tau_1\tau_2 \neq 0$) with convergence rate $\tau_1\tau_2$.

However for the general case (6) such exact computations are no longer possible. Assuming that all Ps are 1 (there are no attacks outside the cycle) the absolute value of its derivative is the product of all the τs of the variables in the cycle: $\prod_{j=1}^{m} \tau_{c_j}$. Because this rule is repeated only after l iterations, the convergence rate should be $\sqrt[l]{\prod_{j=1}^{m} \tau_{c_j}} \leq \sqrt[m]{\prod_{j=1}^{m} \tau_{c_j}}$.

If the assumption of no attacks outside the cycle is dropped, the derivatives cannot be computed as they depend on the P values. Nevertheless we still can argue that asymptotically those values converge and its contribution to the derivative is a positive multiplicative factor smaller than one, decreasing the convergence rate.

For the overall system (1), we expect ISS algorithm to converge linearly with worst case asymptotic convergence rate $\lambda^* = \max_{\langle x_{c_1},\ldots,x_{c_m}\rangle \in C} \sqrt[m]{\prod_{i=1}^{m} \tau_{c_i}}$, where C is the set of all cycles in the graph.

Theorem 21 (Time Complexity) *ISS asymptotic time complexity is in $O(-e/\log\lambda^*)$.*

Proof. Let $\epsilon_k = \|\mathbf{x}^{(k)} - \mathbf{x}^*\|_\infty$. In the case of linear convergence $\epsilon_{k+1} \approx \lambda\epsilon_k$, and since $\lambda\epsilon_k = \lambda^k\epsilon_0$, we have that $\epsilon_{k+1} \approx \lambda^k\epsilon_0$, so $k \approx \log\left(\frac{\epsilon_{k+1}}{\epsilon_0}\right) \log(\lambda)^{-1} \approx -1/\log\lambda^*$. Since computing (5) for every x_i, $1 \leq i \leq n$ requires exactly e steps (each edge in the graph is considered once), the result follows. ∎

5 Experimental Results

Since there are yet no large scale debates of the kind envisioned by Social Abstract Argumentation, in order to assess the performance of the various algorithms as a function of the network structure we generated a set of argumentation graphs consisting in graphs with uniformly distributed vertex degrees. This set partitions into 9 equally sized subsets corresponding to a specific size and density interval combination. Each of these subsets consists of 1000 graphs generated at random with size and density uniformly distributed in the corresponding intervals, amounting to a total of 9000 graphs. Whereas our initial goal was set to deal with 5000 arguments and a *reasonable* number of attacks between them, we generated instances with up to 10000 arguments and densities up to 1. Recall that a density of 1 means a totally connected graph, although we can hardly expect that all arguments attack each other in a real debate. In fact, argument graphs found in the literature or in argumentation websites have small attack densities, often ≤ 0.1. It is expectable that larger debates will have similar attack densities. The node weights τ_i were randomly selected with uniform probability from the interval $]0, 1[$. The initial guess for each variable was set to the corresponding weight: $x_i^{(0)} = \tau_i$, which corresponds to its upper bound.

All algorithms were implemented in C++ and compiled with gcc 4.8. Experiments were performed on an Intel Core i7 CPU @ 2.4 GHz, running Linux 3.12. The tolerance that defines the termination criterion was set to $tol = 10^{-12}$. An algorithm is said to timeout for a given benchmark if it does not converges to the solution with $tol = 10^{-12}$ after 300 seconds.[2]

5.1 ISS, INR, and IFP comparison

The results obtained from running the ISS, IFP, and INR algorithms on the set of benchmarks are summarized in table 1.

	solved	number of times slower than ISS			ISS time
		best	worst	average	
ISS	100%	-	-	-	100%
INR	65%	1	57913	29	16%
IFP	32%	1	233	1.13	0.07%

Table 1. Performance of ISS, INR, and IFP algorithms.

The column "solved" indicates the percentage of graphs that each algorithm was able to solve. It shows that INS and IFP timed out in a significant number of benchmarks. This shows that in practice these algorithms may diverge or have very slow convergence rates (recall that INS and IFP lack theoretical global convergence guarantees).

The subsequent three columns – "best", "worst", "average" – indicate the time ratio between each algorithm and ISS, at best, worst and in the average case among all solved instances. For example, the value 29 in the column "average" means that, on average, INR was slower than ISS by a factor of 29. By comparing runtimes on the fraction of benchmarks solved by these algorithms, we observe that they were never faster than ISS – in fact they may be several orders of magnitude slower. Furthermore, INR is clearly not competitive since it is one order of magnitude slower than ISS on average.

To further characterize the benchmarks, we looked at the time spent by ISS on these instances as a fraction of the total time taken by ISS to solve all the instances ("ISS time" column). This shows the hardness of those instances from the ISS point of view. In particular, we see that IFP is only able to solve the very *easy* benchmarks, i.e. the set of benchmarks where ISS spent 0.07% of the time, showing that IFP does not scale, hence is not a viable alternative either.

5.2 ISS performance analysis

Figure 1 details the performance of the ISS algorithm as a function of the size and density of the argumentation graph on the set of benchmarks. Each chart in the figure plots runtime until convergence.

[2] The implementation of the algorithms and the graphs are available at `http://centria.di.fct.unl.pt/~jleite/iss_graphs`.

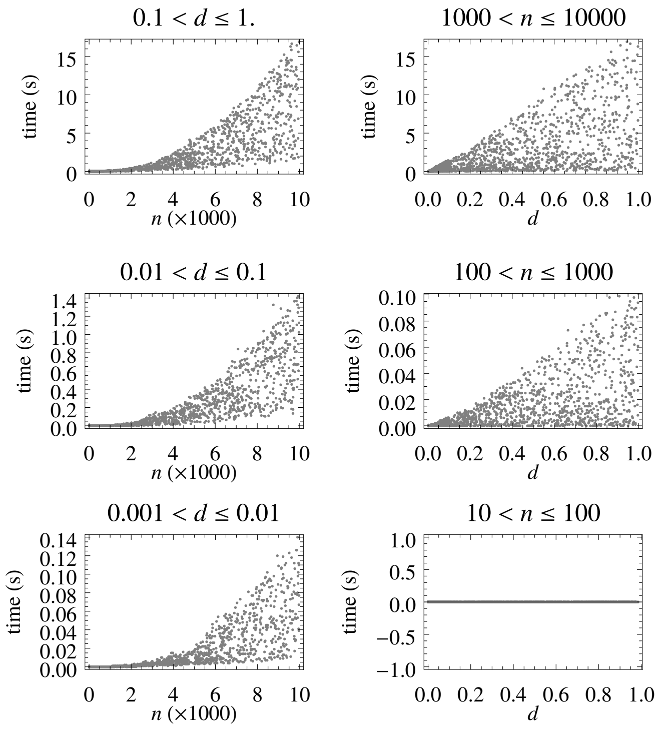

Figure 1. Performance of the ISS algorithm (yy) as function of the size (n) and density (d) of the argumentation graph.

As expected, larger and more dense graphs lead to slower runtimes, while smaller and more sparse graphs are solved very quickly (in particular, all graphs with less than 100 nodes are solved instantaneously). The nonlinear curves observed in the charts that plot runtimes as a function of the network size reflect the fact that the ISS is quadratic in the number of nodes (thm. 21). Similarly, the linear curves of the charts that plot runtimes as a function of the network density are due to ISS being linear in the number of edges.

For the kinds of graphs that we expect to observe in online debate forums, i.e. with $d \leq 0.1$, the ISS algorithm performs within our goal of solving instances with 5000 arguments in under 1 sec. For equally large graphs, though with lower attack densities, we observe extremely fast runtimes in the order of 0.1 sec. Note that largest runtime observed was in the order of 15 seconds for fully connected graphs with 10000 nodes, which is still quite remarkable.

6 Exploiting the Debate Structure

Whereas the graphs generated in the previous section have uniform distributed vertex degrees, which often result in one strongly connected component i.e. all arguments are involved in a cycle with every other argument, this is not what is expected in reality. Even by looking at small existing debates, we see that there are clusters of strongly connected components (like mini debates to settle on some claim) that are arranged as a tree. The system described in [2] even goes further and restrict the debate to a tree of arguments, which is adequate given the restricted domain of application, though too restrictive for large scale web debates.

In this section we improve the ISS algorithm to take advantage of the topological structure of the debate, namely if the groups of interconnected variables form a tree.

Algorithm 22 (ISS^c) *Let* $\mathbf{S} = \left(S_1, \ldots, S_{|\mathbf{S}|}\right)$ *denote the topologically ordered sequence of strongly connected components (sccs) of* *the argumentation graph. For each $1 \leq k \leq |\mathbf{S}|$, call ISS for the subsystem of equations (1) restricted to the x_i where $i \in S_k$.*

We can easily show that ISS^c retains global convergence.

Theorem 23 (Global Convergence) *ISS^c always converges to a solution of system (1) independently of the initial guess $\mathbf{x}^{(0)} \in]0,1[^n$.*

Proof. Since $\forall_{i \in S_1} A_i \subseteq S_1$ then ISS converges for the system $\cup_{i \in S_1} x_i$ (thm.19). By prop.13 we are left with an equivalent but smaller system (all variables $x_i : i \in S_1$ were replaced by x_i^*). The iteration of this process for all the remaining sccs guarantees the convergence of the original system. ■

Furthermore, the time complexity is better (or equal if there is a single scc) than with ISS.

Theorem 24 (Time Complexity) *ISS^c asymptotic time complexity is in $O\left(-c\,|\mathbf{S}|\,/\log\lambda^*\right)$, where c is the number of edges of the ssc with the largest number of edges.*

Proof. Finding a topologically ordered sequence of sccs in a directed graph is in $O\left(n+e\right)$, and calling ISS on any scc of the graph is in $O\left(-c/\log\lambda^*\right)$. ■

Corollary 25 *ISS^c is optimal when the graph is a tree.*

Proof. If the argumentation graph is a tree then ISS^c calls ISS n times, with an accumulated cost of $O\left(e\right)$. Note that factor $-1/\log\lambda^*$ vanishes since there are no cycles in the graph. ■

To assess the performance ISS^c on the average case, we have solved all benchmarks of section 6 and compared the runtimes with those obtained by ISS. Additionally, to measure the cost of the preprocessing phase (i.e. finding the topologically ordered sequence of strongly connected components) we also considered the runtimes obtained by algorithm ISS^c less the preprocessing time (denoted ISS^{c-}). The results are summarized in figure 2.[3] Not only there is a substantial gain in using ISS^c, but the gain in practice is closer to the gain observed when ISS^{c-} is used, since the partition of the graph in the sccs can be done incrementally as the debate progresses.

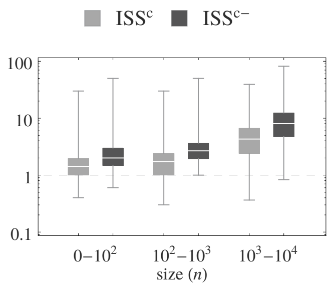

Figure 2. Average number of times that ISS^c and ISS^{c-} are faster than ISS.

7 Conclusions

In this paper we presented the Iterative Successive Substitution (ISS) to efficiently approximate the models of debates structured according to *Social Abstract Argumentation* [10]. We have shown it to be effective in the sense that, unlike other iterative algorithms, it always converges, and efficient as it can quickly deal with systems with thousands of arguments. Additionally, with a negligible overhead, it can increase performance by taking advantage of the topological structure of the debate graphs.

The *Iterative Successive Substitution* algorithm opens up the possibility to effectively and efficiently develop an online debating tool such as the one envisioned in [10]. But its significance goes well beyond that. First, the semantics of Social Abstract Argumentation can be employed in existing online argumentation systems, such as argunet.org, debate.org, debategraph.org, agora.gatech.edu, yourview.org.au cohere.open.ac.uk, among others, either to directly assign meaning to the combination of the argumentation graph and votes, which exist but are not taken into account in a combined way, or by providing an effective way to extend their systems with votes.

Other systems such as MicroDebates [8] can also directly employ this semantics and implementation, addressing one potential concern with the fact that, in their current system, each debate has several models which may be difficult to grasp by the common user. Systems such as those described in [9] and [5], aimed at extracting arguments and make sense of natural language debates could also benefit from the combination of the graph structure and the numerical values associated with the votes that the *Iterative Successive Substitution* makes possible to use in practice. The application described in [2] can also benefit from this algorithm by being able to drop the restriction to have debates shaped as trees only.

Finally, other systems which are characterized by similar non-linear systems of equations [3, 11] can benefit from this algorithm.

Our immediate future work draws on these observations, and will focus on finding out exactly how relevant ISS is outside the realm of Social Argumentation, develop an application for Social Abstract Argumentation, and investigate the applicability in the context of the above mentioned systems. All this appears to be very fruitful terrain.

ACKNOWLEDGEMENTS

We would like to thank Martin Aleksandrov for his help with some preliminary experiments. João Leite was partially supported by FCT under project ASPEN (PTDC/EIA-CCO/110921/2009).

REFERENCES

[1] I. K. Argyros and F. Szidarovszky, *The Theory and Applications of Iteration Methods*, Systems Engineering, Taylor & Francis, 1993.
[2] P. Baroni, M. Romano, F. Toni, M. Aurisicchio, and G. Bertanza, 'An argumentation-based approach for automatic evaluation of design debates', in *CLIMA XIV*, volume 8143 of *LNCS*. Springer, (2013).
[3] H. Barringer, D. M. Gabbay, and J. Woods, 'Temporal dynamics of support and attack networks: From argumentation to zoology', in *Mechanizing Math. Reasoning*, volume 2605 of *LNCS*. Springer, (2005).
[4] H. Barringer, D. M. Gabbay, and J. Woods, 'Temporal, numerical and meta-level dynamics in argumentation networks', *Argument & Computation*, **3**(2-3), 143–202, (2012).
[5] E. Cabrio and S. Villata, 'A natural language bipolar argumentation approach to support users in online debate interactions;', *Argument & Computation*, **4**(3), 209–230, (2013).
[6] S. Egilmez, J. Martins, and J. Leite, 'Extending social abstract argumentation with votes on attacks', in *TAFA'13*, volume 8306 of *LNCS*. Springer, (2014).
[7] D. M. Gabbay, 'Equational approach to argumentation networks', *Argument & Computation*, **3**(2-3), 87–142, (2012).
[8] S. Gabbriellini and P. Torroni, 'Large scale agreements via microdebates', in *AT'12*, volume 918 of *CEUR Workshop Proceedings*, (2012).
[9] K. Grosse, C. I. Chesñevar, and A. G. Maguitman, 'An argument-based approach to mining opinions from twitter', in *AT'12*, volume 918 of *CEUR Workshop Proceedings*, (2012).
[10] J. Leite and J. Martins, 'Social abstract argumentation', in *IJCAI'11*. AAAI/IJCAI, (2011).
[11] N. Madrid and M. Ojeda-Aciego, 'On the existence and unicity of stable models in normal residuated logic programs', *Int. J. of Computer Mathematics*, **89**(3), 310–324, (2012).
[12] J. M. Ortega and W.C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[3] The band inside each box is the median, and the bottom and top of each box correspond to the first and third quartiles. The lower and upper ends of the whiskers mark the minimum and maximum of the data.