
The Equational Approach to CF2 Semantics

DOV M. GABBAY

BAR ILAN UNIVERSITY, ISRAEL;

KING'S COLLEGE LONDON, UK;

UNIVERSITY OF LUXEMBOURG, LUXEMBOURG

PAPER 459H: DOVPAPERS/459O/459-EACF2S.TEX

FEBRUARY 2012; VERSION DATED 15 FEBRUARY 2012

ABSTRACT. We introduce a family of new equational semantics for argumentation networks which can handle odd and even loops in a uniform manner. We offer one version of equational semantics which is equivalent to CF2 semantics, and a better version which gives the same results as traditional Dung semantics for even loops but can still handle odd loops.

1 Background and orientation

Our starting point is the important papers of Baroni, Giacomin and Guida on odd loops and SCC recursiveness [1, 6]. In their papers the authors offer the CF2 semantics in response to difficulties arising from the Dung semantics handling of odd and even loops. In our paper we outline our equational approach to argumentation networks and show how the CF2 semantics can be obtained from perturbations to the equations associated with the networks. This approach will offer additional methodological support for the CF2 semantics, while at the same time show the power of the equational approach. We offer our own loop-busting equational semantics LB, which includes CF2 as a special case.

The structure of this paper is as follows. Section 2 reproduces the motivating discussion from [1] for the CF2 semantics and points out its weaknesses. Section 3 introduces the equational semantics. Section 4 defines our loop busting semantics LB. Section 5 introduces our semantics LB2 and compares with CF2 on the technical level. We conclude with a general discussion in Section 6.

2 CF2 semantics as introduced in the SCC paper [1]

Baroni *et al.* devote a long discussion about the inadequacy of the traditional semantics in handling odd and even loops. They say, and I quote:

“the length of the leftmost cycle should not affect the justification states [of an argument]. More generally, it is counter-intuitive that different results in conceptually similar situations depend on the length of the cycle. Symmetry reasons suggest that all cycles should be treated equally and should yield the same results.”

We now reproduce Figure 8 of [1], and discuss the problems associated with it.

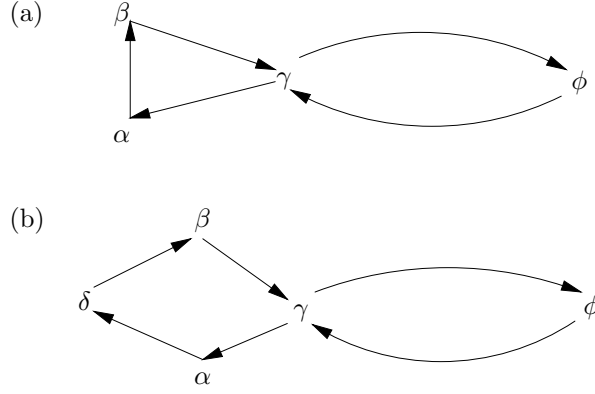


Figure 1. Figure 8 of [1]. Problematic argumentation of frameworks

The only preferred extension for Figure 1(a) is $\{\phi, \alpha\}$, while for Figure 1(b) we have the extensions $\{\alpha, \beta, \phi\}$ and $\{\delta, \gamma\}$. These two results are conceptually different, in (a) ϕ is not prevented from being justified while in (b) it is prevented.

A more striking problem is the one outlined in Figure 9 of Baroni *et al.* [1], here reproduced as Figure 2.

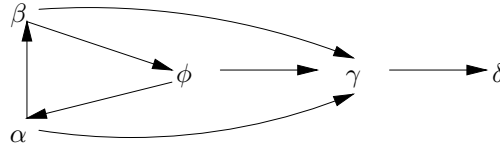


Figure 2. Figure 9 of [1]: Floating defeat and floating acceptance

The only extension in traditional Dung semantics is all undecided. Common sense, however, expects γ to be out and δ to be in. In [5], this is characterised as “one of the main unsolved problems in argumentation-based semantics”.

The CF2 semantics of [1] treats the loops of Figures 1(a) and 1(b) and 2 all in the same way, by taking as CF2 extensions maximal conflict-free sets. We therefore get for Figure 1(a) the CF2 extensions

$$\{\alpha, \phi\}, \{\beta, \phi\} \text{ and } \{\gamma\}$$

and for Figure 1(b) we get

$$\{\alpha, \beta, \phi\}, \{\delta, \gamma\} \text{ and } \{\delta, \phi\}$$

and for Figure 2 we get the extensions

$$\{\beta, \delta\}, \{\alpha, \delta\} \text{ and } \{\phi, \delta\}.$$

Let us put forward a figure of our own, Figure 3. This is a 9 point cycle. The CF2 semantics will take all maximal conflict free subsets as extensions, including among them $\{a_3, a_6, a_9\}$ and its cyclic translations as well as $\{a_1, a_3, a_5, a_7\}$ and its cyclic translations (e.g. $\{a_2, a_4, a_6, a_8\}$, etc.).

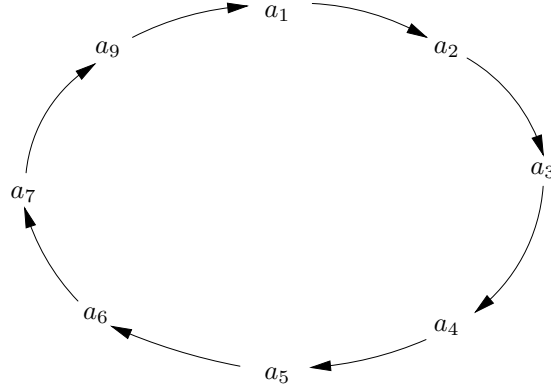


Figure 3.

We shall see later that some of our loop busting semantics LB yield only $\{a_1, a_3, a_5, a_7\}$ and its cyclic translations and not $\{a_3, a_6, a_9\}$, but other LB semantics does yield it.

We agree with [1] on the need for a new approach but we feel that the CF2 semantics offered as a solution requires further independent methodological justification. The notion of conflict freeness is a neutral notion and does not use the central notion of “attack” of the Dung semantics. When we get a loop like $\{\alpha, \beta, \gamma\}$, in a real life application as in Figure 1(a), there are good reasons for the loop in the context of the application area where it arises, and we want a decisive solution to the loop in terms of $\{\text{in}, \text{out}\}$, which makes sense in the application area. We do not want just a technical, non-decisive choice of maximal conflict free sets, a sort of compromise which involves no real decision making. Imagine we have a loop with $\{\alpha, \beta, \gamma\}$, and we go to a judge and we expect some effective decision making. We hope for something like “I think γ is not serious”.

Taking the maximal conflict free sets in this case, namely $\{\alpha\}$, $\{\beta\}$ and $\{\gamma\}$ means nothing. We would perceive that the judge is not doing his job properly and that he is just offering us options which are obvious and non-controversial, given the geometry of the loop! See [3] for extensive examples of resolving loops in a practical realistic way.

Another problem, in our opinion, with the CF2 semantics is that it is an overkill as far as loop-breaking is concerned. If we look at Figures 1(a) and 1(b) and replace them by Figure 4 and 5 our loop-breaking needs are the same according to [1], but in Figure 4, we do not need the extensions $\{a_3, a_6, a_9, b\}$ from the loop-breaking point of view. In our LB semantics we do not mind if there will be less extensions than CF2, in the odd cycle case of Figure 4 but insist that there will be the same extensions as the traditional Dung semantics

in the even cycle of Figure 5. CF2 gives more extensions then the traditional Dung extensions for Figure 5.

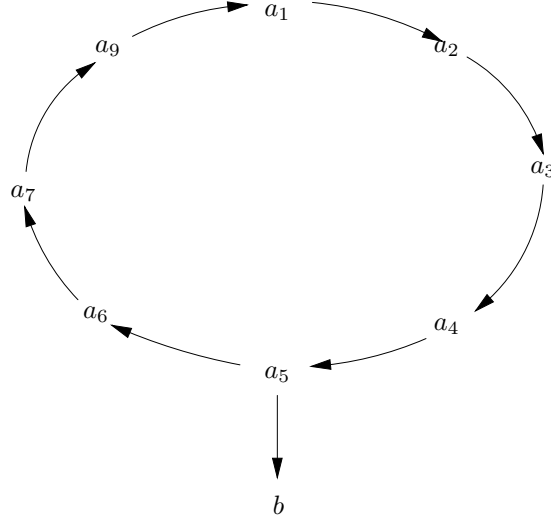


Figure 4.

We should realise that within the context of argumentation theory alone, the maximal conflict free CF2 solution seems somewhat arbitrary, a device which is just technically successful.

It is also the only device available for the loop breaking in this context.

The next sections will discuss the equational approach of [4], and introduce the new LB semantics.

3 The equational approach

Let $\mathcal{A} = (S, R)$ be an argumentation frame $S \neq \emptyset$ is the set of arguments and $R \subseteq S \times S$ is the attack relation. The equational approach views (S, R) as a bearer of equations with the elements of S as the variables ranging over $[0, 1]$ and with R as the generator of equations. Let $x \in S$ and let y_1, \dots, y_k be all of its attackers. We write two types of equations $Eq_{\max}(\mathcal{A})$ and $Eq_{\text{inverse}}(\mathcal{A})$.¹

For Eq_{\max} we write

- $x = 1 - \max(y_1, \dots, y_k)$
- $x = 1$ if it has no attackers.

For Eq_{inverse} we write

- $x = \prod_{i=1}^k (1 - y_i)$
- $x = 1$, if it has no attackers.

¹In [4] there are more Eq options.

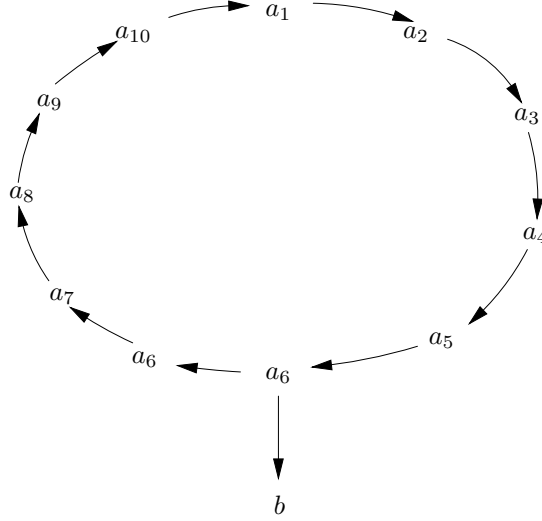


Figure 5.

We seek solutions \mathbf{f} for the above equations. In [4] we prove the following:

THEOREM 1.

1. *There is always at least one solution in $[0, 1]$ to any system of continuous equations $Eq(\mathcal{A})$.*
2. *If we use $Eq_{\max}(\mathcal{A})$ then the solutions \mathbf{f} correspond exactly to the Dung extensions of A . Namely*
 - $\mathbf{f}(x) = 1$ *corresponds to* $x = in$
 - $\mathbf{f}(x) = 0$ *corresponds to* $x = out$
 - $0 < \mathbf{f}(x) < 1$ *corresponds to* $x = undecided$.

The actual value in $[0, 1]$ reflects the degree of odd looping involving x .
3. *If we use Eq_{inverse} , we give more sensitivity to loops. For example the more undecided elements y attack x , the closer to 0 (out) its value gets.*

In the context of equations, a very natural step to take is to look at *Perturbations*. If the equations describe a physical or economic system in equilibrium, we want to change the solution a bit (perturb the variables) and see how it affects the system. For example, when we go to the bank to negotiate a mortgage, we start with the amount we want to borrow and indicate for how many years we want the loan and then solve equations that tell us what the monthly payment is going to be. We then might change the amount or the number of years or even negotiate the interest rate if we find the monthly payments too high.

In the equational system arising from an argumentation network we can try and fix the value of some arguments and see what happens. In the equational context, this move is quite natural. We shall see later, that fixing some values to 0 in the equations of $Eq(\mathcal{A})$, amounts to adopting the CF2 semantics, when done in a certain way. When done in other ways it gives the new loop-busting semantics LB.

EXAMPLE 2. Consider Figure 2. The equations for this figure are (we use Eq_{inverse})

1. $\alpha = 1 - \phi$
2. $\beta = 1 - \alpha$
3. $\phi = 1 - \beta$
4. $\gamma = (1 - \alpha)(1 - \beta)(1 - \phi)$
5. $\delta = 1 - \gamma$

The solution here is

$$\begin{aligned}\alpha &= \beta = \phi = \frac{1}{2} \\ \gamma &= \frac{1}{8} \\ \delta &= \frac{7}{8}\end{aligned}$$

Let us perturb the equation by adding an external force which makes a node equal zero. The best analogy I can think of is in electrical networks where you make the voltage of a node 0 by connecting it to earth.

Let $Z(x)$ be the “earth” connection for node x . We now do several perturbations as examples

- (a). Let's choose to make $\phi = 0$.

We replace equation 3 by

$$3^*a. \phi = (1 - \beta)Z(\phi)$$

$$3^*b. Z(\phi) = 0.^2$$

The equations now solve to

$$\begin{aligned}\phi &= 0, \alpha = 1, \beta = 0 \\ \gamma &= 0 \\ \delta &= 1.\end{aligned}$$

This gives us the extension $\{\alpha, \delta\}$

- (b). If we try to make $\alpha = 0$, we replace equation (1) by

²We use Z and write 3^*a and 3^*b , rather than just writing $3^* a = 0$ because of algebraic considerations. The current equations can be manipulated algebraically to prove $a = b = c$. By adding a fourth variable $Z(\phi)$ we prevent that.

$$1^*a. \alpha = (1 - \phi)Z(\alpha)$$

$$1^*b. Z(\alpha) = 0$$

We solve the equations and get

$$\alpha = 0, \beta = 1, \phi = 0$$

$$\gamma = 0$$

$$\delta = 1$$

This corresponds to the extension $\{\beta, \delta\}$.

(c). Now let us make $\beta = 0$. We replace equation (1) by

$$2^*a. \beta = (1 - \alpha)Z(\beta)$$

$$2^*b. Z(\beta) = 0$$

Solving the new equations gives us

$$\beta = 0, \phi = 1, \alpha = 0$$

$$\gamma = 0$$

$$\delta = 1$$

This gives us the extension $\{\phi, \delta\}$.

If we compare these extensions with the CF2 extensions, we see that they are the same.

EXAMPLE 3. Let us see what happens with Figure 1(b). Here we have a well behaved even loop. Let us write the equations

$$1. \alpha = 1 - \gamma$$

$$2. \delta = 1 - \alpha$$

$$3. \beta = 1 - \delta$$

$$4. \gamma = 1 - \beta$$

$$5. \phi = 1 - \gamma$$

Let us do some perturbations:

(a) Let us make $\gamma = 0$. We change equation 4 to

$$4^*a. \gamma = (1 - \beta)Z(\gamma)$$

$$4^*b. Z(\gamma) = 0$$

We solve the new equations and get

$$\gamma = 0, \alpha = 1, \delta = 0, \beta = 1, \phi = 1.$$

The extension is $\{\alpha, \beta, \phi\}$.

(b) Let us try $\alpha = 0$. we replace equation 1 by

$$1^*a. \alpha = (1 - \gamma)Z(\alpha)$$

$$1^*b. Z(\alpha) = 0$$

We solve the new equations and get

$$\alpha = 0, \delta = 1, \beta = 0, \gamma = 1, \phi = 0$$

The extension we get is $\{\delta, \gamma\}$.

(c) Let us make $\delta = 0$. We replace equation 2 by

$$2^*a. \delta = (1 - \alpha)Z(\delta)$$

$$2^*b. Z(\delta) = 0$$

The solution is

$$\delta = 0, \alpha = 1, \beta = 1, \gamma = 0 \text{ and } \phi = 1$$

This gives the extension

$$\{\alpha, \beta, \phi\}$$

(d) Let us make $\beta = 0$. the new equations for β are

$$3^*a. \beta = (1 - \delta)Z(\beta)$$

$$3^*b. Z(\beta) = 0$$

We solve the new set of equations and get

$$\beta = 0, \gamma = 1, \phi = 0, \alpha = 0, \delta = 1.$$

The extension is $\{\gamma, \delta\}$.

(e) Let us make $\phi = 0$. We change equation 5 to

$$5^*a. \phi = (1 - \gamma)Z(\phi)$$

$$5^*b. Z(\phi) = 0$$

We solve the new equations.

From (3) and (4) we get

$$5. \delta = \gamma$$

From (1) and (2) we get

$$7. \alpha = \beta.$$

Let $\alpha = \beta = x$. Then $\gamma = \delta = 1 - x$.

If we want $\{0, 1\}$ extensions, i.e. $x \in \{0, 1\}$, then we get the extensions

$\{\alpha, \beta\}$, case $\{x = 1, \phi = 0\}$

$\{\gamma, \delta\}$, case $\{x = 0, \phi = 0\}$.

(f) Let us make $\alpha = \gamma = 0$. The new equations are

$$1^*a. \alpha = (1 - \gamma)Z(\alpha)$$

$$1^*b. Z(\alpha) = 0$$

$$2. \delta = 1 - \alpha$$

$$3. \beta = 1 - \delta$$

$$4^*a. \gamma = (1 - \beta)Z(\gamma)$$

$$4^*b. Z(\gamma) = 0$$

$$5. \phi = 1 - \gamma.$$

The solution is

$$\alpha = \gamma = 0$$

$$\delta = 1$$

$$\beta = 0$$

$$\phi = 1$$

The extension we get is $\{\delta, \phi\}$.

(g) Let us summarise in Table 1.

Case	Set B of points made 0	Corresponding extensions
(a)	$B_a = \{\gamma\}$	$\{\alpha, \beta, \phi\}$
(b)	$B_b = \{\alpha\}$	$\{\delta, \gamma\}$
(c)	$B_c = \{\delta\}$	$\{\alpha, \beta, \phi\}$
(d)	$B_d = \{\beta\}$	$\{\gamma, \delta\}$
(e)	$B_e = \{\phi\}$	$\{\alpha, \beta\}, \{\gamma, \delta\}$
(f)	$B_f = \{\alpha, \gamma\}$	$\{\delta, \phi\}$

Table 1.

4 The equational loop-busting semantics LB for complete loops

We now introduce our loop busting semantics, the LB semantics for complete loops. We need a series of concepts leading up to it.

DEFINITION 4 (Loops). Let $\mathcal{A} = (S, R)$ be an argumentation network.

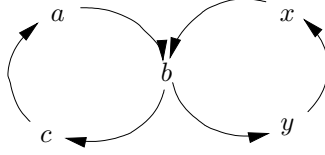


Figure 6.

1. A subset $E = \{x_1, \dots, x_n\} \subseteq S$ is a loop cycle, (or a loop set, or a loop) if we have

$$x_1 R x_2, x_2 R x_3, \dots, x_{n-1} R x_n, x_n R x_1$$

(S, R) is said to be a complete loop if every element of S is an element of some loop cycle.³

2. A set $B \subseteq S$ is a loop-buster if for every loop set E we have $E \cap B \neq \emptyset$
3. Let $B \subseteq S$ be a loop-buster and let \mathcal{M} be a meta-predicate describing properties of B . We can talk about the semantics LBM , where, (when we define it later), we use only loop-busters B such that $\mathcal{M}(B)$ holds. Criteria for adequacy for LBM are
 - (a) It busts all odd numbered loops
 - (b) It busts all even numbered loops and yields all allowable Dung extensions for such loops.
4. Our first two proposals for conditions \mathcal{M} on loop-busters is minimality. The idea is the smaller B is, the more options we have. Therefore, we define: A loop-buster set B is minimal absolute if there is no loop-buster set B' with a smaller number of elements (we do not require $B' \subseteq B$!).
5. A loop-buster set B is minimal relative if there does not exist a $B' \subsetneq B$ which is a loop-buster set.

EXAMPLE 5 (Loop-buster 1). Consider Figures 6 and 7.

1. In Figure 6 there are two loop sets, $\{a, b, c\}$ and $\{b, x, y\}$. The loop-buster $\{b\}$ is minimal absolute and $\{y, c\}$ is minimal relative. The loop set $\{y, b\}$ is not minimal absolute.
2. Consider Figure 7. There are two loops $\{a, b, c\}$ and $\{a, b, c, x\}$. The minimal absolute loop-buster sets are $\{c\}, \{a\}$. $\{x, b\}$ is not minimal relative.

EXAMPLE 6 (Loop-buster 2).

Consider Figure 8.

The loops in this figure are many. For example, we list some

³Comparing with the terminology of [1], a complete loop is a union of disjoint strongly connected sets.

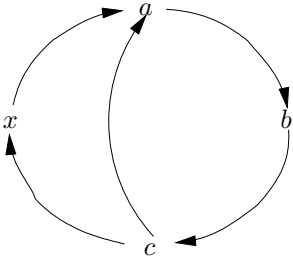


Figure 7.

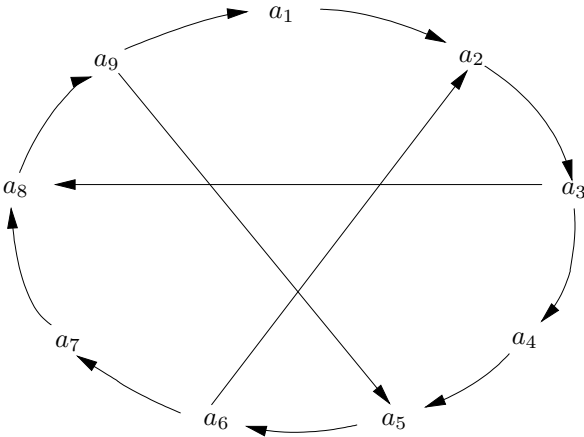


Figure 8.

1. $\{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9\}$
2. $\{a_6, a_2, a_3, a_4, a_5\}$
3. $\{a_3, a_8, a_9, a_1, a_2\}$
4. $\{a_9, a_5, a_6, a_7, a_8\}$

Consider the loop-buster

$$\{a_2, a_5, a_8\}$$

This is not a minimal absolute set but if we delete one of its elements we get a minimal absolute set. No one element is a loop-buster.

DEFINITION 7 (The loop-busting semantics LBM for complete loops). Let $\mathcal{A} = (S, R)$ be an argumentation network. Assume that (S, R) is a complete loop, namely that each of its elements belongs to some loop cycle, as defined in item 1 of Definition 4. We define the LBM extensions for \mathcal{A} as follows.

1. Let B be a loop-buster for \mathcal{A} satisfying \mathcal{M} .
2. Let $E_{q_{\max}}(\mathcal{A})$ be the system of equations generated by \mathcal{A} . These have the form

$$(\mathbf{eq}(x)) : x = \mathbf{h}_x(y_1, \dots, y_{k(x)})$$

where $x \in S$, and $y_1, \dots, y_{k(x)}$ are all the attackers of x . If x has no attackers then $h_x \equiv 1$.

3. For each $x \in B$ replace the equation $\mathbf{eq}(x)$ by the two new equations

- $(\mathbf{eq}_a^*(x)) : x = \mathbf{h}_x(y_1, \dots, y_{k(x)})Z(x)$
- $(\mathbf{eq}_b^*(x)) : Z(x) = 0$

where $Z(x)$ is a new variable syntactically depending on x alone.

4. Solve the equations in (3) and let \mathbf{f}_B be any solution.

Then the set

$$E_{\mathbf{f}, B} = \{x \in S \mid \mathbf{f}_B(x) = 1\}$$

is an LBM extension.

5. Thus the set of all LBM extensions for $\mathcal{A} = (S, R)$ is the set

$$\{E_{\mathbf{f}, B} \mid B \text{ is as in (1), } \mathbf{f}_B \text{ is as in (4) and } E_{\mathbf{f}, B} \text{ is as in (4)}\}$$

Note that our definition of extension for a general network will be given in the next section.

Before we prove soundness of LBM relative to the traditional Dung semantics and compare LBM with CF2 semantics, let us do some examples. We use Figures 2 and 1(b).

EXAMPLE 8. Consider Figure 2. The only loop here is $\{\alpha, \beta, \phi\}$. There are three minimal absolute loop-busting sets, $B_\alpha = \{\alpha\}$, $B_\beta = \{\beta\}$ and $B_\phi = \{\phi\}$.

For each one of these sets we need to modify the equations of Figure 2 and solve them and see what extensions we get. This has already been done in Example 2, parts (a), (b) and (c).

In (a) we made $\phi = 0$, i.e. we used the loop-busting set B_ϕ . We solved the modified equations and got the extension $\{\alpha, \delta\} = E_\phi$. In (b) we made $\alpha = 0$, i.e. we used the set B_α , solved the modified equations and got the extension $E_\alpha = \{\beta, \delta\}$.

In (c) we made $\beta = 0$, i.e. we used the set B_β , solved the modified equations and got the extension $E_\beta = \{\phi, \delta\}$.

Let us now compare with the CF2 extensions for the figure (Figure 2). The maximal conflict free sets of the first loop $\{\alpha, \beta, \phi\}$ are $C_\alpha = \{\alpha\}$, $C_\beta = \{\beta\}$ and $C_\phi = \{\phi\}$. They are the same as our loop-busting sets, but they are used differently. They are supposed to be in (i.e. value 1) not out (value 0). We use $C_\alpha, C_\beta, C_\phi$ to calculate the CF2 extensions and get $\{\alpha, \delta\}$, $\{\beta, \delta\}$ and $\{\phi, \delta\}$, indeed the same as the LB extensions.

EXAMPLE 9. We now consider Figure 1(b). The only minimal absolute loop-buster set here is $B_\gamma = \{\gamma\}$. We have three more minimal relative sets, $B_1 = \{\beta, \phi\}$, $B_2 = \{\delta, \phi\}$ and $B_3 = \{\alpha, \phi\}$.

We refer the reader to Example 3, where some equational calculations for this figure are carried out.

1. In (a) of Example 3, we make $\gamma = 0$, we solve the modified equation and get the extension $E_\gamma = \{\alpha, \beta, \phi\}$.

This takes care of the case $B_\gamma = \{\gamma\}$.

2. Let us address the case of $B_3 = \{\alpha, \phi\}$. We use (b) of Example 3, where we make $\alpha = 0$. We modify the equation for α and get a solution $\alpha = 0, \delta = 1, \beta = 0, \gamma = 1$ and $\phi = 0$.

We needed to also make $\phi = 0$ for the loop-buster set $\{\alpha, \phi\}$, but as it turns out, making $\alpha = 0$ also makes $\phi = 0$. We thus get the extension $E_{\alpha, \phi} = \{\delta, \gamma\}$.

3. Let us address the case of $B_1 = \{\beta, \phi\}$. This corresponds to case (d) $\beta = 0$ of Example 3. We modify the equations and solve them and get $\beta = 0, \gamma = 1, \phi = 0, \alpha = 0$ and $\delta = 1$.

The extension is $\{\gamma, \delta\}$.

Again, although we did not explicitly make the requirement $\phi = 0$, the equations obtained from the requirement $\alpha = 0$ did the job for us.

4. We now check the case of $B_2 = \{\delta, \phi\}$. Here we get a discrepancy with case (c) of Example 3.

There, in case (c), we only require $\delta = 0$, solve the equations and get the extension $\{\alpha, \beta, \phi\}$. This is not what we want, as we also require $\phi = 0$. So let us do the calculation in detail here.

The modified equation system for $B_1 = \{\delta, \phi\}$ is the following:

1. $\alpha = 1 - \gamma$

- 2*a. $\delta = (1 - \alpha)Z(\delta)$
- 2*b. $Z(\delta) = 0.$
- 3. $\beta = 1 - \delta$
- 5. $\gamma = 1 - \beta$
- 5*a. $\gamma = \phi = (1 - \gamma)Z(\phi)$
- 5*b. $Z(\phi) = 0.$

We solve the equations and get $\phi = 0, \delta = 0, \beta = 1, \gamma = 0, \alpha = 1.$

The extension is $\{\alpha, \beta\}.$

EXAMPLE 10 (CF2 and the LB minimal absolute semantics). The LB minimal absolute semantics does not give all the CF2 extensions in the case of even loops. Consider Figure 5. The set $B = \{a_{10}\}$ yields the extension $E_B = \{b, a_1, a_3, a_5, a_7, a_0\}.$ B is minimal absolute. Consider now B' being $B' = \{a_{10}, a_3, a_6\}.$

This yields

$$E_{B'} = \{a_1, a_4, a_7, a_9\}$$

However, B' is not minimal absolute. $E_{B'}$ is a CF2 extension. B' is a minimal relative set.

What happens here is that the minimal absolute semantics gives the same extensions for even loops as the traditional Dung extensions, but the CF2 semantics gives more. This is a weakness of CF2.

REMARK 11 (CF2 and the minimal relative extensions). Let us discuss the results of Example 9 calculated for Figure 1(b) and compare them with the CF2 extensions of Figure 1(b). This will give us an idea about the relation of CF2 to the minimal relative semantics. We use Example 3, where all the extensions were calculated and especially refer to Table 1, given in item (g) of Example 3, which summarises these calculations.

1. The CF2 extensions are all the conflict free subsets. These are $\{\delta, \phi\}, \{\delta, \gamma\}, \{\alpha, \beta, \phi\}.$

Comparing with the semantics of Table 1, we get the following: the LB minimal absolute extensions are one only, namely $\{\delta, \gamma\}.$ The LB minimal relative extensions are $\{\delta, \gamma\}, \{\alpha, \beta, \phi\}$ and $\{\alpha, \beta\}, \{\gamma, \phi\}.$

We see that LB minimal absolute gives less extensions (but breaks loops) while LB minimal relative gives one more extension. Obviously we need to identify a policy \mathcal{M} which will yield exactly the CF2 extensions.

2. Let us examine case (4) of Example 9 more closely. This is the case of $B_2 = \{\delta, \phi\}$ of Figure 1(b). The loop-buster set B_2 was introduced to bust two loops. The loop $\{\alpha, \beta, \gamma, \delta\}$ and the loop $\{\gamma, \phi\}.$ δ was included to bust the first loop and γ was included to bust the second loop. Our equational computations show in case (c) of Example 3 that if we start with $\delta = 0$ we get that it follows that $\gamma = 0.$ But γ belongs also to the second loop $\{\gamma, \phi\}.$ So $\{\delta\}$ on its own is a loop-buster for both loops and we do not need to include ϕ in the loop-buster. So $B_1 = \{\delta, \phi\}$

is not minimal relative because $B'_2 = \{\delta\}$ can do the job. The above considerations show that the definition of minimal relative loop-busting sets needs to be adjusted. This needs to be done in a methodologically correct manner and will be addressed in the next section.

Note that if we accept that $B'_2 = \{\delta\}$ is the minimal relative loop-busting set, then the calculated extension for this case is $E_\delta = \{\delta, \phi\}$, in complete agreement with the CF2 semantics!

We now need to demonstrate the soundness of the LB semantics. The perceptive reader will ask himself, how do the LB extensions relate to the extensions of traditional Dung semantics? After all, we start with the standard equational semantics, which for the case of Eq_{\max} is identical with the Dung semantics, but then using a loop-busting set B of one kind or another, we get a new set of equations and call the solutions LB extensions. What are these solutions and what meaning can we give them?

Obviously, we need some sort of soundness result. This is the job of the next theorem.

THEOREM 12 (Representation theorem for LB semantics). *Let $\mathcal{A} = (S, R)$ be an argumentation net being a complete loop as in Definition 7 and let B be a loop-busting subset of S (of some sort \mathcal{M}). Let $\mathbf{E}(B, \mathcal{A})$ be the family of LB extensions obtained from \mathcal{A} and B by following the procedures of Definition 7. Then $\mathbf{E}(B, \mathcal{A})$ can be obtained also following the procedure below*

1. For each $x \in B$, let $\mathbf{z}(x)$ be a new point not in S . Let $\mathbf{z}(x)$ be all different for different x s.
2. Define (S_B, R_B) as follows:

$$\begin{aligned} S_B &= S \cup \{\mathbf{z}(x) | x \in B\} \\ R_B &= R \cup \{(\mathbf{z}(x), x) | x \in B\}. \end{aligned}$$

3. The network (S_B, R_B) is an ordinary Dung network and has traditional Dung extensions. We have (for Eq_{\max}):

$$\mathbf{E}(B, \mathcal{A}) = \{E \cap S | E \text{ is an extension of } (S_B, R_B)\}$$

Proof. The new equations for each $x \in B$ in (S_B, R_B) are

$$\begin{aligned} (\mathbf{eq}_a^*(x)) : x &= 1 - \max(y_1, \dots, y_{k(x)}, \mathbf{z}(x)) \\ (\mathbf{eq}_a^*(\mathbf{z}(x))) : \mathbf{z}(x) &= 1 \end{aligned}$$

where $y_1, \dots, y_{k(x)}$ are all the attackers of x in (S, R) .

Since $\mathbf{z}(x) = 1$, we get that

$$\begin{aligned} 1 - \max(y_2, \dots, y_{k(x)}, \mathbf{z}(x)) &= (1 - \max(y_1, \dots, y_{k(x)})(1 - \mathbf{z}(x))) \\ &= (1 - \max(y_1, \dots, y_{k(x)}))\mathbf{z}(x) \end{aligned}$$

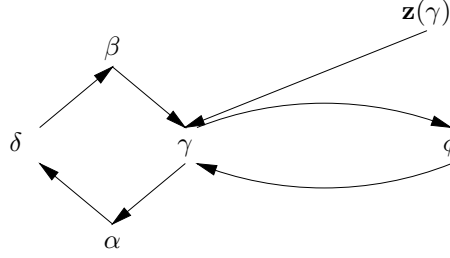


Figure 9. Case (1): $\gamma = 0$ for $B_\gamma = \{0\}$

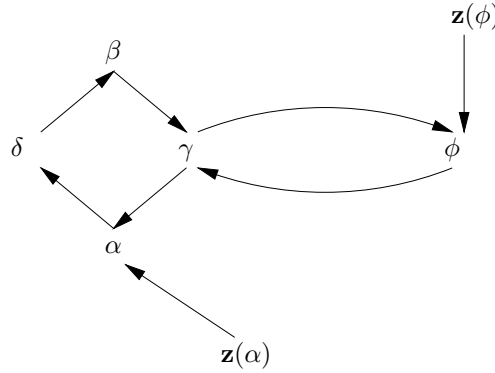


Figure 10. Case (2): $B_3 = \{\alpha, \phi\}$

provided $Z(x) = 1 - \mathbf{z}(x)$.

Of course $\mathbf{z}(x) = 1$ means $Z(x) = 0$.

So we get the same modified equations as required by the LB semantics in Definition 7. ■

EXAMPLE 13. Let us represent the cases of Example 9, which dealt with Figure 1(b). See Figures 9, 10, 11, 12 corresponding to cases (1)–(4) of Example 9.

We are also adding Figure 13, describing the situation for $B'_2 = \{\delta\}$ as discussed in Remark 11 in item (b).

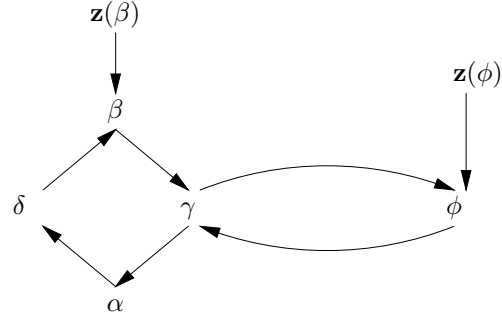
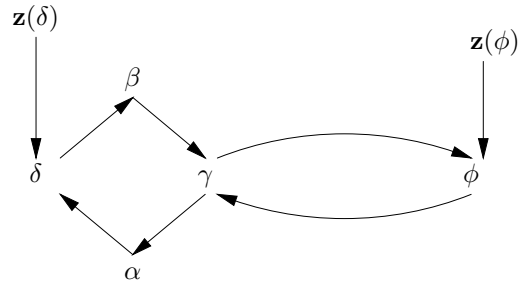
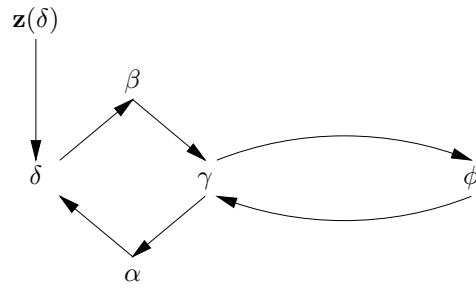
5 The equational semantics LB and its connection with CF2

We now define the family of LB semantics and identify the loop-busting counterpart of CF2. We need to develop some concepts first. We begin with a high school example.

EXAMPLE 14 (High school example).

1. Solve the following equations in the unknowns x, y, z .

(a) $x - y = 1$

Figure 11. Case (3): $B_1 = \{\beta, \phi\}$ Figure 12. Case (4): $B_2 = \{\delta, \phi\}$ Figure 13. Case (b): $B'_2 = \{\delta\}$

- (b) $x + y = 5$
(c) $z^2 - 4yz + x + 1 = 0$

The point I want to make is that we solve the equations directionally. We first find the values of x and y from equations (a) and (b) to be $x = 3$ and $y = 1$ and then substitute in equation (c) and solve it. We get

- (c) $z^2 - 4z + 4 = 0$
 $z = 2$

2. Let us change the problem a bit. We have the equations

- (a) $x - \sin y = 2.99$
(b) $x + \sin y = 3.01$
(c) $z^2 - 400yz + x + 1 = 0$

Here we may again consider equations (a) and (b) first but also use the approximation $y \approx \sin y$. We find $x = 3, y \approx 0.01$ and solve the third to get $z = 2$.

3. A third possibility is to look at equations (a) and (b) and decide to ignore them altogether,⁴ and substitute $x = y = 0$. We get

- (c*). $z^2 + 1 = 0$

4. Another example is the equation

$$x^4 - 2x^2 + \frac{x}{1000} + 1 = 0.$$

To solve this equation we decide on the perturbation which ignores $\frac{x}{1000}$ on account of it being relatively small. We solve

$$x^4 - 2x^2 + 1 = 0$$

we get $x = \pm 1$.

REMARK 15. We present a perturbation protocol for solving equations of the form

$$x = \mathbf{h}_x(v_1, \dots).$$

1. Let V be a set of variables and \mathbb{E} be a set of equations of the form $x = \mathbf{h}_x(V_x)$, where $V_x \subseteq V$ are the variables appearing in \mathbf{h}_x , and x ranges over V . We seek solutions to the system \mathbb{E} with values hopefully in $\{0, 1\}$. If \mathbf{h}_x are all continuous functions in $[0, 1]$, then we know that there are solutions with values in $[0, 1]$, but are there solutions with values in $\{0, 1\}$?

Even if we are looking for and happy with any kind of solution, we may wish to shorten the computation by starting with some good guesses, or

⁴Of course, ignoring (a) and (b) needs to be justified.

some approximation or follow any kind of protocol \mathbb{P} which will enable us to perturb the equations and get some results which we would find satisfactory from the point of view of our application area.

In the case of equations arising from argumentation networks, we would like perturbations which help us overcome odd-numbered loops.

Note that in numerical analysis such equations are well known. If x_1, \dots, x_m are variables in $[0, 1]$ and $\mathbf{h}_1, \dots, \mathbf{h}_m$ are continuous functions in $[0, 1]$, we want to solve the equations

$$x_i = \mathbf{h}_i(x_1, \dots, x_m), i = 1, \dots, m.$$

One well known method is that of successive approximations. We guess a starting value

$$x_1 = a_1^0, \dots, x_m = a_m^0$$

and continue by substituting

$$a_i^{j+1} = \mathbf{h}_i(a_1^j, \dots, a_m^j).$$

Under certain conditions on the functions \mathbf{h}_i (Lipschitz condition), the values $a_i^j, j = 1, 2, \dots$ converge to a limit $a_i^\infty, i = 1, 2, \dots, m$ and that would be a solution. What we are going to do in this paper is in the same spirit.

2. Let us proceed formally adopting a purely equational point of view and take a subset $B_1 \subseteq V$ of the variables and decide for our own reasons to substitute the value 0 for all the variables in B_1 in the equations \mathbb{E} .

How we choose B_1 is not said here, we assume that we have some protocols for finding such a B_1 . In the application area of argumentation, these protocols will be different loop-busting protocols $\text{LB}(\mathcal{M})$.

For the moment, formally from the equational point of view, we have a set of equations \mathbb{E} with variables V and a $B_1 \subseteq V$, which we want to make 0. How do we proceed?

This has to be done carefully and so we replace for each $u \in B_1$, the equation

$$\mathbf{eq}(u) : u = \mathbf{h}_u(V_u)$$

by the pair of equations

$$\mathbf{eq}^*(u) : \begin{cases} u = \mathbf{h}_u(V_u)Z(u) \\ Z(u) = 0 \end{cases}$$

We now propagate these values through the new set of equations, solve what we can solve and end up with new equations of the form

$$\mathbf{eq}^1(x) : x = \mathbf{h}_x^1(V_x^1)$$

for $x \in V$, where \mathbf{h}_x^1 is the new equation for x and V_x^1 are its variables. We have

$$V_x^1 \subseteq V - B_1.$$

The variables of B_1 get all value 0 and maybe more variables solve to some numerical values. Note that we can allow also for the case of $B_1 = \emptyset$.

We always have a solution because the functions involved are all continuous.

Let $U_1 = \{x | V_x^1 = \emptyset\}$. U_1 is the set of x which get a definite numerical value, for which V_x , the set of variables they depend on, is empty. We have $B_1 \subseteq U_1 \subseteq V$.

Let \mathbf{f}_1 be a function collecting these values on U_1 , i.e. $\mathbf{f}(x) = \mathbf{h}_x^1$, for $x \in U_1$.

3. We refer to U_1 as the set of all elements instantiated to numerical values at step 1. We declare all variables of U_1 as having rank 1.

4. Let \mathbb{E}^1 be the system of equations for the variables in $V - U_1$.

We now have a new system of variables and we can repeat the procedure by using a new set B_2 chosen to make 0.

We can carry this procedure repeatedly until we get numerical values for all variables. Say that at step n we have that the union of all sets U_1, U_2, \dots, U_n equals V . Then also each element of V has a clear rank k , the step at which x was instantiated. Call this procedure Protocol $\mathbb{P} = (B_1, B_2, \dots)$. Note that we did not say why and how we choose the sets B_1, B_2, \dots . In the case of equations arising from argumentation networks, these sets B_i will be loop-busting sets.

5. Note that the equations initially give variables either 0 or 1 and our loop busters also give variables 0, and Eq_{\max} and Eq_{inverse} are such that they keep the variables in $\{0, 1\}$, then all the functions \mathbf{f} involved are $\{0, 1\}$ functions

EXAMPLE 16. We now explain why we use Z in our perturbation. Consider the equations

1. $a = 1 - c$
2. $b = 1 - a$
3. $c = 1 - b$.

These equations correspond to a 3-element argumentation loop.

We take the $B_1 = \{a\}$ and want to execute a perturbation. If we do just substitute $a = 0$, we get a contradiction because the equations prove algebraically through manipulation that

$$a = b = c = 1 - a = 1 - b = 1 - c$$

So we need to change the equation governing a . We write

$$1^*a. \quad a = (1 - c)Z(a)$$

$$1^*b. \quad Z(a) = 0$$

Algebraically we now have 4 equations in 4 variables

$$a, b, c, Z(a)$$

The solution is

$$Z(a) = 0, a = 0, b = 1, c = 0.$$

We cannot any more execute an algebraic manipulation to get $a = b = c$!

EXAMPLE 17. Let us recall Example 2, manipulating the equations arising from Figure 2. This is an illustration of our procedure. We used the loop-busting sets $B_\alpha = \{\alpha\}$, $B_\beta = \{\beta\}$ and $B_\phi = \{\phi\}$, and followed the procedure as described in Remark 15.

Let us now proceed with more concepts leading the way to the full definition of our loop-busting LB semantics.

We saw how to get a set of equations $Eq(\mathcal{A})$ from any argumentation network \mathcal{A} . Now we want to show how to get an argumentation network $\mathcal{B}_\mathbb{E}$ from any set of equations \mathbb{E} .

Furthermore, once we have a set of equations $\mathcal{B}_\mathbb{E}$, we can perturb it to get a new set of equations \mathbb{E}_B using some perturbation set B and then from the equations \mathbb{E}_B get a new argumentation network $\mathcal{A}_{\mathbb{E}_B}$. The net result of all these steps is that we start with a network $\mathcal{B} = (S, R)$ and a perturbation set of nodes $B \subseteq S$ and we end up with a new network which we can denote by $\mathcal{A} = \mathcal{B}_B$. If B is a loop-busting set, then \mathcal{A} is the loop-busted result of applying B to \mathcal{B} .

DEFINITION 18.

1. Let V be a set of variables and let $x = \mathbf{h}_x(V)$, $x \in V$, $V_x \subseteq V$ be a system of equations \mathbb{E} , where V_x is the set of variables actually appearing in \mathbf{h}_x . We now define the associated argumentation network $\mathcal{A}_\mathbb{E} = (S_\mathbb{E}, R_\mathbb{E})$ as follows:

- (a) Let $S_\mathbb{E} = V$
- (b) Let $yR_\mathbb{E}x$ hold iff $y \in V_x$.⁵

⁵The definition of $yR_\mathbb{E}x$ as $y \in V_x$ is a very special definition, making essential use of the fact that the equation

$$x = \mathbf{h}_x(V_x)$$

is of a very special form of either

$$x = 1 - \max V_x$$

or

$$x = \prod_{y \in V_x} (1 - y)$$

The real definition, which is more general, should be

$$yR_\mathbb{E}x \text{ iff when we substitute } y = 1 \text{ in } \mathbf{h}_x, \text{ we get that } \mathbf{h}_x = 0.$$

This definition is good in a more general context.

Suppose y_1, y_2 attack x *jointly*. This means that $x = 0$ only if both $y_1 = y_2 = 1$. See [9] for a discussion of joint attacks.

The equation for that is

$$x = 1 - y_1 y_2$$

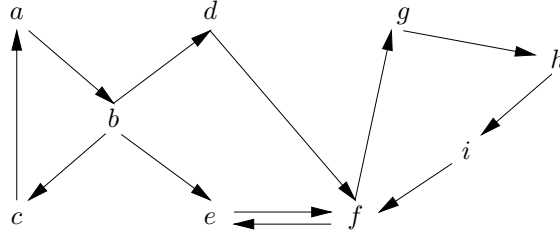


Figure 14.

2. Let $\mathcal{B} = (S, R)$ be a network and let $\mathbb{E} = Eq(\mathcal{B})$ be its system of equations.

\mathbb{E} is a system of equations as in (a) above. Let $B \subseteq V$ be some of the variables in V . Let \mathbf{f} be a function giving numerical values 0 to the variables in B .

Let $\mathbb{E}_{\mathbf{f}}$ be the system of equations obtained from \mathbb{E} by substituting the values $\mathbf{f}(u)$ in the equations for the variables of B . The variables of $\mathbb{E}_{\mathbf{f}}$ are $V - B$. Consider now the argumentation network

$$\mathcal{A}_{\mathbb{E}_{\mathbf{f}}} = (S_{\mathbb{E}_{\mathbf{f}}}, R_{\mathbb{E}_{\mathbf{f}}}).$$

We say that $\mathcal{A}_{\mathbb{E}_{\mathbf{f}}}$ was derived from \mathcal{B} using \mathbf{f} . We can also use the notation $\mathcal{B}_{\mathbf{f}}$ or \mathcal{B}_B .

EXAMPLE 19. Let us use the network of Figure 14 to illustrate the process outlined in Remark 15. This figure is used extensively in [2] and also quoted in [3].

The variables of this figure are

$$V = \{a, b, c, d, e, f, g, h, i\}$$

The equations are, using Eq_{inverse} as follows:

1. $a = 1 - c$
2. $b = 1 - a$
3. $c = 1 - b$
4. $d = 1 - b$
5. $e = (1 - b)(1 - f)$

Given a general equation

$$x = \mathbf{h}_x(V_x)$$

for example

$$x = \mathbf{h}_x(y_1, y_2, z) = (1 - z)(1 - y_1, y_2)$$

We define the notion for $V_x^0 \subseteq V_x$ of joint attack as follows.

V_x^0 attack x jointly if the substitution of $u = 1$ for all variables in V_x^0 makes $\mathbf{h}_x = 0$ and for no proper subset of V_x^0 do we have this property.

So in the above example, y_1, y_2 attack x jointly and z attacks x singly.

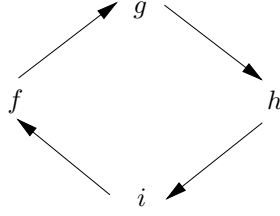


Figure 15.

$$6. f = (1 - e)(1 - d)(1 - i)$$

$$7. g = (1 - f)$$

$$8. h = (1 - g)$$

$$9. i = (1 - h)$$

Let us take $B = \{a\}$ and let \mathbf{f} be the function making $a = 0$ (i.e. $\mathbf{f}(a) = 0$). (This is a loop-busting move, breaking the loop $\{a, b, c\}$).

The new equations for a are

$$1^*a. \quad a = (1 - c)Z(a)$$

$$1^*b. \quad Z(a) = 0$$

or we can simply write

$$1^*. \quad a = 0$$

Substituting this value in the equations and solving we get the new system of equations for the unknown variable as follows

$$1. \quad b = 1, \text{ known value}$$

$$2. \quad c = 0, \text{ known value}$$

$$3. \quad d = 0, \text{ known value}$$

$$4. \quad f = 1 - i$$

$$5. \quad g = 1 - f$$

$$6. \quad h = 1 - g$$

$$7. \quad i = 1 - h$$

We get the solution function \mathbf{f}_1 giving the known values to the variables $a = 0, b = 1, c = 0, d = 0, e = 0$ (these are the variables of rank 2) and the new system of equations (6), (7), (8), (9). Using item (1) of Definition 18, we get the derived network in Figure 15.

We can continue now with this loop and choose a loop-busting variable say $B' = \{i\}$. We substitute $i = 0$ in the equations and get $f = 1, g = 0, h = 1, i =$

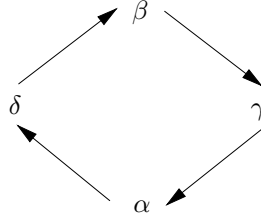


Figure 16.

0 (these are the variables of rank 1). We extend the function \mathbf{f}_1 to be \mathbf{f}_2 giving these values.

We thus get the extension $\{b, f, h\}$ (these are the variables which get value 1 from \mathbf{f}_2). We also get clear ranks for the variables for the particular protocol $\mathbb{P} = (B, B') = (\{a\}, \{i\})$.

EXAMPLE 20. Let us do another example. We use Figure 1 item b. We note that in Example 3, item (e), we make $\phi = 0$. This means we start with $B_1 = \{\phi\}$.

We manipulated the equations in item (e) of Example 3 and got the remaining equation (1)–(4), namely

1. $\alpha = 1 - \gamma$
2. $\delta = 1 - \alpha$
3. $\beta = 1 - \delta$
4. $\gamma = 1 - \beta$

We proceeded in item (e) of Example 3 to find two solutions to these equations. However, if we follow the procedures of Remark 15, we need now to extract a new network out of equations (1)–(4), and keep in mind the partial function $\mathbf{f}_1(\phi) = 0$. The new network is presented in Figure 16.

We can now proceed by choosing a new loop-busting set B_2 . We have four options here. Choosing B_2 to be α or β will make the extension $\alpha = \beta = 0, \delta = \gamma = 1$ and choosing B_2 to the γ or δ will give the extension $\alpha = \beta = 1, \gamma = \delta = 0$.

These are also the solutions we got in item (e) of Definition 3.

We now want to define procedures which will give us the CF2 extensions of Baroni *et al.*. We need to give a protocol of which B to use, as outlined in Definition 18.

DEFINITION 21.

1. Let $\mathcal{A} = (S, R)$ be an argumentation network. Define $x \approx y$ iff $x = y$ or for some loop $u_1 R u_2, u_2 R u_3, \dots, u_n R u_1$ we have $x, y \in \{u_1, \dots, u_n\}$. This is an equivalence relation on S . Let S^* be a set of equivalence

classes.⁶ Define R^* on S^* by $x^* R^* y^*$ iff for some $x' \approx x, y' \approx y$ we have $x' R y'$.

(S^*, R^*) is an ordering without loops.

2. Let $S_x \subseteq S$ be an \approx equivalence class of $x \in S$. We say that S_x is top-class if the following holds

- $y R z \wedge z \approx x \rightarrow y \approx x$.

The above means that no other disjoint class S_y attacks any member of S_x .

Let $C \subseteq S_x$ be a maximal subset of conflict free points in S_x . Let $B = S_x - C$. Then using B in the protocol of Remark 15 shall be referred to as *using the CF2 protocol*.

LEMMA 22. *In the notation of Definition 21, if we apply the protocol of Remark 15, we get that all elements of B solve to 0 and all elements of C solve to 1.*

Proof. Let x be an element of C . The equation for x is

$$x = \mathbf{h}_x(V_x)$$

where $V_x = \{y | y \text{ attacks } x\}$.

\mathbf{h}_x can be the expression $1 - \max\{y | y \in V_x\}$ according to Eq_{\max} or $\prod_{y \in V_x} (1 - y)$ for resp. Eq_{inverse} or any other choice as long as the following condition holds.

- If for all $y \in V_x, y = 0$ then $\mathbf{h}_x(V_x) = 1$.

The important point to note is that since S_x is a top loop, all attackers of x are in S_x , and since $x \in C$ and C is a *maximal* conflict free set, all attackers of x are not in C , so they are in B and so they are 0. Hence $x = 1$.⁷

The above consideration shows that B can represent C , and so Baroni *et al.* conflict free choice C can be represented as our equational loop-busting set B . We now continue our protocol with B and get a new network \mathcal{A}_B (note we are using the notation of Definition 21 and our original network was $\mathcal{A} = (S, R)$.) ■

REMARK 23. Let $\mathcal{A} = (S, R)$ be a network and suppose it does contain points that are not attacked. If we choose $B = \emptyset$ and apply our procedure of Remark 15, what do we get?

The procedure solves the equations as much as possible to get a new network \mathcal{A}_1 and a function \mathbf{f}_1 giving numerical values nodes which are in or out in the grounded extension. In other words, they have a $\{0, 1\}$ numerical value. Thus what we get is the rest of the network after the grounded extension has been eliminated.

⁶These are the maximal strongly connected sets in the terminology of [1].

⁷Obviously our proof does not work unless C is maximal conflict free set. So the LB semantics can yield the CF2 semantics only because Baroni *et al.* chose to take *maximal* conflict free set. I shall ask Baroni for his reasons for this choice.

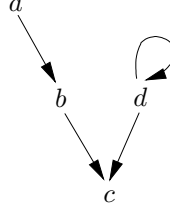


Figure 17.



Figure 18.

EXAMPLE 24. Consider the network of Figure 17. Begin the procedure with $B_1 = \emptyset$. We get $\mathbf{f}_1(a) = 1, \mathbf{f}_1(b) = 0$ and the remaining network \mathcal{A}_1 of Figure 18.

We can now continue with the loop-busting set $B_1 = \{d\}$, follow the procedure for \mathcal{A}_1 and get $d = 0, c = 1$. Thus the solution with the loop buster sets $B_1 = \emptyset, B_2 = \{d\}$ yields the function $\mathbf{f}_2(a) = 1, \mathbf{f}_2(b) = 0, \mathbf{f}_2(d) = 0, \mathbf{f}_2(c) = 1$.

The corresponding extension is $\{a, c\}$.

REMARK 25.

1. We are now in a position to define our $\text{LB}\mathcal{M}$ extensions. We see from the discussions so far that we can use the following procedure to find $\{0, 1\}$ functions \mathbf{f} on an argumentation network $\mathcal{A}_0 = (S_0, R_0)$.

Step 1.

Define $\mathcal{A}_0^* = (S_0^*, R_0^*)$ as in Definition 21. Choose a loop-busting subset B_1 for the top loops of \mathcal{A}_0^* , following some meta-level considerations, i.e. satisfying \mathcal{M} .

Step 2.

Apply the procedure of Remark 15, using B_1 and \mathcal{A}_0 and get \mathbf{f}_1 and \mathcal{A}_1 . Recall that following the terminology of Remark 15 item 3, all elements in $S_0 - S_1$ have rank 1. These are the elements instantiated to numerical values at Step 1. Furthermore \mathbf{f}_1 is a $\{0, 1\}$ function. Also note that $\mathcal{A}_1 = (S_1, R_0 \cap S_1)$.

Step 3.

Choose a new loop-busting set B_2 for the top loops of \mathcal{A}_1 using our meta-level considerations \mathcal{M} .

Step 4.

Go to step 2 to \mathcal{A}_1 using B_2 and obtain \mathbf{f}_2 and \mathcal{A}_2 .

Also identify the elements of $S_2 - S_1$ as the elements of rank 2.

Step $n + 2$.

Continue until you get $\mathcal{A}_{n+3} = \emptyset$.

The function \mathbf{f}_{n+2} will be total on S_0 and will give you the extension

$$E(B_1, B_2, \dots, B_{n+2}) = \{x | \mathbf{f}_{n+2}(x) = 1\}.$$

All elements in the network have a clearly defined rank, it being the step in which they were instantiated to numerical value in $\{0, 1\}$.

2. We define the semantics LBM as the family of all the extensions of the form $E(B_1, \dots, B_{n+2})$, for all possible choices of B_i allowable by \mathcal{M} .

We now want to proceed to define our loop-busting semantics LB1, LB2, LB3, and LB4. We want to use the notions of minimal absolute and minimal relative. We already remarked in item (2) of Remark 11 that the above notions need to be adjusted. We now have the tools to do so.

DEFINITION 26 (Computational loop-busting set).

1. Let (S, R) be a complete loop, as defined in Definition 4. Let B_1 be a subset of S . We say that B_1 is a computational loop-buster if, when we follow the procedure outlined in Remark 15, and get the function \mathbf{f}_1 as described there, we have that

$$B'_1 = \{x | \mathbf{f}_1(x) = 0\}$$

is a loop-busting set (according to Definition 4), namely B'_1 intersects every loop cycle in S .

Note that what this definition says is very simple. There is no need for B_1 itself to intersect every loop C . Making all points in B_1 equal 0 in the equations would make more points 0, namely we get B'_1 and B'_1 does intersect every loop. If this is the case, we say that B_1 is a computational loop-buster.

2. A computational loop-buster is minimal absolute if there is no smaller computational loop buster with a smaller number of elements.

B_1 is a minimal relative computational loop-buster, if no proper subset of it is such.

DEFINITION 27. We define the following loop-busting semantics.

1. LB1

We are allowed to choose only loop-busting sets which contain elements from top loops (as defined in Definition 21, item (1)), and which are computationally minimal absolute (as defined in Definition 26).

So in other words, if our network is (S, R) and S_x, S_y, \dots are all the top loops then our set B is a subset of $S_x \cup S_y \cup \dots$ and it is a computational minimal absolute set for any loop in any top loop.

2. LB2 (to be proved equivalent to the CF2 semantics)
We are allowed to choose only loop-busting sets obtained from maximal conflict free subsets of top loops, as defined in Definition 21, item (2).
3. LB3
We are allowed to choose only loop-busting sets containing elements from top loops (as defined in Definition 21, item (1)), and which are computational minimal relative (as defined in Definition 26).
4. LB4 (Directional Shkop semantics of [3])
We are allowed to choose loop-busting sets containing elements from top loops (as defined in Definition 21, item (1)) and which are a single loop element.

So this semantics busts top loops one at a time.

Note that the LB4 semantics gives rise to the same extensions as LB3. The reason is that if we start with a computationally minimal relative set B , we can substitute its elements one by one following the protocol of LB4 semantics and we never bust all loops until we substitute the last element, because the set is minimal relative. But now we are doing an equivalent LB4 semantics!

We now show that our LB2 semantics is the same as CF2.

The following is a definition of CF2 extensions, as given in [2, Definition 4]. The notion of strongly connected set used was defined in Definition 21.

DEFINITION 28. Let $\mathcal{A} = (S, R)$ be an argumentation network and let $E \subseteq S$ be a set of arguments, then E is a CF2 extension of \mathcal{A} iff

1. If (S, R) itself is a strongly connected set then E is a maximal conflict free subset of S .
2. Otherwise, for every C where C is a maximal strongly connected subset of S , we have that the set $C \cap E$ is a CF2 extension of the network (T_1^C, R_1) , where

$$\begin{aligned} T_1^C &= C - \{x | \exists y \in E((y, x) \in R \wedge y \notin C)\} \\ R_1 &= R \cap (T_1^C \times T_1^C). \end{aligned}$$

THEOREM 29 ($LB2 = CF2$). *The semantics CF2 is the same as the semantics LB2.*

Proof.

1. We start by showing that every LB2 extension E of a network $\mathcal{A}_0 = (S_0, R_0)$ is also a CF2 extension as defined in Definition 28. To achieve this goal we need to follow closely how the extension E was defined in LB2 for \mathcal{A}_0 .

The LB2 semantics was defined in Definition 27 by using the protocols of Remark 25. The loop-busting sets involved in these protocols were defined in item 2 of Definition 21.

Let us list the way the LB2 extension E of \mathcal{A}_0 is defined.

- (a) E is defined according to item 2 of Remark 25. The extension is obtained in the form $E = E(B_1, \dots, B_{n+2})$, where each B_{i+1} is a loop-buster on the top loops of the network $\mathcal{A}_i = (S_i, R_i)$.
- (b) The loop buster was chosen according to the protocol (CF2 protocol) of item 2 of Definition 21.
- (c) The elements of $S_i - S_{i+1}$ are of rank i , where i is the step in which they got a numerical value in $\{0, 1\}$ by the function \mathbf{f}_i .
The function \mathbf{f}_{n+1} gives numerical values in $\{0, 1\}$ to all the elements of S_0 and we have

$$E = \{x \in S_0 \mid \mathbf{f}_{n+1}(x) = 1\}.$$

We are now going to use the rank i to show that E is a CF2 extension according to Definition 28. We need a bit more preparation.

- (d) Let $\mathcal{A}_0^* = (S_0^*, R_0^*)$ be the ordering without loop derived from (S_0, R_0) as in Definition 21. The element classes of S_0^* are all the maximal strongly connected subsets of S_0

We continue the proof by induction on $n + 2$, being the number of steps required to define E .

- (e) Case $n + 2 = 2(n = 0)$.

In this case we have that (S_0, R_0) itself is a strongly connected set. Then E is obtained from B_1 which satisfies the CF2 condition as in Step 1 of Remark 25. Using the notation of Step 1, we have $E = \{x \mid \mathbf{f}_1(x) = 1\}$. In this case E is also a CF2 extension.

- (f) Case $n > 0$

Take any strongly connected maximal subset C of (S_0, R_0) . Let $k + 1$ be the step at which all elements of C get a numerical value. There are two possibilities:

- i. At step k some maximal subset $C' \subseteq C$ does not yet have numerical values. C' is a top loop in \mathcal{A}_k . In this case B_k is a loop buster for C' and makes it get a numerical value in \mathcal{A}_{k+1} .
- ii. C is not a top loop in \mathcal{A}_k , in which case B_k busts some other loops and in the process of obtaining \mathcal{A}_{k+1} , C' disappears as all these elements get a numerical value. In fact, in this case it is Step k which gives all elements of C a numerical value.

Let us now look at the set T_1^C , as defined in item 2 of Definition 28.

$$T_1^C = C - \{\exists y \in (E - C)((y, x) \in R)\}$$

The set T_1^C is comprised from two parts, $T_{1,k}^C$ and $T_{1,k+1}^C$. Part $T_{1,k}^C$ are all points $z \in T_1^C$ that get numerical value at step k by \mathbf{f}_k and the set $T_{1,k+1}^C$ is the set of all points that get numerical values at step $k + 1$ by the function \mathbf{f}_{k+1} .

In case (i) above, $T_{1,k+1}^C$ is still the loop C' , but still $T_{1,k}^C$ may be $\neq \emptyset$.

In case (ii), $T_{1,k+1}^C = \emptyset$. We ask is $E \cap T_1^C$ a CF2 extension of T_1^C according to Definition 28? The answer is yes. The part $T_{1,k}^C$ is calculated traditionally and if there is a loop $C' = T_{1,k+1}^C$, it will be busted by B_k which was chosen in LB2 to yield a maximal conflict free set.

We thus see through considerations (a)–(e) that $\text{LB2} \subseteq \text{CF2}$.

The reader should see Remark 30, to appreciate the difference between the way LB2 and CF2 calculate their extensions.

2. We now prove the other direction, namely that every CF2 extension is also an LB2 extension.

Let E_0 be a CF2 extension of the network $\mathcal{A}_0 = (S_0, R_0)$. We would like to define a sequence of LB2 loop-busters B_1, B_2, \dots, B_{n+2} such that $E_0 = E(B_1, \dots, B_{n+2})$. We choose B_i by looking at E_0 .

Step 1.

Look at the top loops of (S_0, R_0) . Use E_0 to choose the loop-busters.

Let us look at top strongly connected sets of (S_0, R_0) . These are either single unattacked points x for which the LB2 equation is $x = 1$ (in agreement with CF2) or a loop C , for which CF2 gives a choice of maximal conflict-free subset $E_0 \cap C$. We can now choose our LB loop-buster B_1 to be

$$B_1 = \bigcup_{\text{top loops } C \text{ of } \mathcal{A}_0} (C - E_0).$$

We now apply step 1 of the LB2 procedure and get $\mathcal{A}_1 = (S_1, R_1)$. Again consider the top loops of \mathcal{A}_1 . Let

$$B_1 = \bigcup_{\text{top loops } C \text{ of } \mathcal{A}_1} (C - E_0)$$

We carry on in this manner and get

$$E(B_1, B_2, \dots).$$

To show that $E_0 = E$ is not difficult. This is done along the lines of the proof of (1) above.

■

REMARK 30. The way we compute the extensions of LB2 are not synchronised with the way CF2 itself works. This can be seen from the network of Figure 14. In this figure the top loop is $\{a, b, c\}$. By choosing the maximal conflict free set $\{c\}$, we are led to the loop-buster $B = \{b\}$.

In step 1 we propagate the attacks (solve the equations) to get numerical values for as many variables as we can.

We get in LB2

$$a = 0, c = 1, b = 0, d = 1, e = 1, f = 0, g = 1, h = 0 \text{ and } i = 1.$$

We get the extension in one step

$$E = \{c, d, e, g, i\} = E(\{b\}).$$

In comparison, when we follow the CF2 procedures, we look at two loops, the strongly connected sets, $\{a, b, c\}$ and $\{e, f, g, h, i\}$.

Step 1 of the LB2 procedure corresponds to what the CF2 definition does, namely treat the loop $\{a, b, c\}$. This we do by choosing $c = 1$ and calculating $a = 0, b = 0$ and $d = 1$.

We now look at the loop $\{e, f, g, h, i\}$ and take from it the elements attacked from outside it. In this case we take the element f attacked by d . Thus we are left with $S_1 = \{e, g, h, i\}$ and R_1 being $\{(g, h), (h, i)\}$. The CF2 extension for (S_1, R_1) is $\{e, g, i\}$. So the extension we get finally is $E = \{c, d, e, g, i\}$.

The intuition of LB2 is to solve equations and get numerical values as much as possible. In the process we bust loops, by making loop variables equal 0. Because we are dealing with equations, we can make loop variables equal 0 one by one.

CF2 in comparison, is different.

1. It concentrates more on loops
2. It treats loops by choosing maximal conflict free sets and makes them 1.
3. Since it is not dealing with equations, it must make 1 batches of variables (maximal conflict free sets) and cannot do them one by one. So LB2 has the same problem. It must make 0 batches of variables all in one go.

See [2] for an algorithm for CF2. I think they do it inductively as we did in this example.

REMARK 31 (Computational complexity). The computational complexity of the LB semantics is daunting. This is because of the requirement that the loop-busting sets in LB1 and LB3 be computationally minimal. This means that we have to solve many equations before we can choose our sets. So I don't regard LB1 and LB3 as practical. In comparison, LB4 is easy, we just choose a loop element based on the geometry of the network and proceed recursively. This is simple and easy.

Two factors are in our favour. One mathematical and one social.

The mathematical one is that since we are dealing with equation solving, there are tools available for our use such as MATHEMATICA or MATLAB or MAPLE or NSolve which can help.

We note that there is actually no need to go to equations at all, in view of the soundness results in the representation Theorem 12.

We can work completely with argumentation networks.

On the social side, our argumentation community is blessed with many talented young researchers such as Sarah Gaggl and Stefan Woltran (see [2]) and

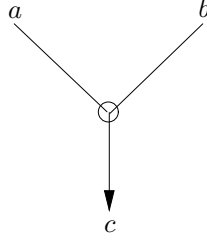


Figure 19.

Wolfgang Dvorak, who write good algorithms, and who (should they take an interest in the LB semantics) would certainly find a way around complexity difficulties.

Perhaps the comparison in Remark 30 shows the way of how the algorithms of [2] can be adapted to the LB semantics.

REMARK 32 (Comparison of LB1 semantics with the CF2 semantics). Let us make a quick comparison.

1. LB1 give the correct exact extension for even loops. CF2 gives more extensions.
2. LB1 does bust odd loops but gives less extensions than CF2.
3. LB1 remains conceptually within the attack concept framework. It has the backing of hundreds of years of experience in approximate/guess/simplification of equations. CF2 uses the slightly out of sync concept of maximal conflict free sets, and although LB2 gives it an equational flavour, it is forced and not natural (see next item 4).
4. Most importantly, and this has been pointed out to me by Martin Caminada, CF2 is not robust against conceptual extensions, such as joint attacks. This is shown in the next remark, 33.

We shall also see that the LB semantics is robust.

REMARK 33 (CF2 and joint attacks).

1. In my paper [9] of Fibring argumentation frames, I introduced the notion of joint attack of say two arguments a and b on a third argument c . This means that c is in only when both a and b are out. I used the notation of Figure 19.

The equation for c is

$$c = 1 - ab.$$

I also showed in the paper how to interpret joint attacks within ordinary argumentation networks, using for each node e the new auxiliary points $x(e)$ and $y(e)$. The joint attack of Figure 19 can be represented faithfully by Figure 20.

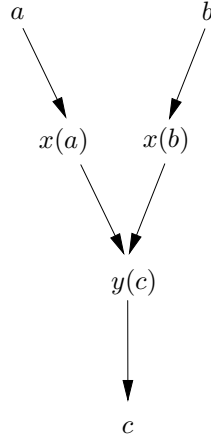


Figure 20.

It is important to note that if we calculate the equations of Figure 20 we get for Eq_{inverse}

$$x(a) = 1 - a$$

$$x(b) = 1 - b$$

$$y(c) = ab$$

$$c = 1 - ab$$

If we use Eq_{max} we get

$$x(a) = 1 - a$$

$$x(b) = 1 - b$$

$$y(c) = 1 - \max(1 - a, 1 - b)$$

$$c = \max(1 - a, 1 - b) = 1 - \min(a, b)$$

It is clear that in either case the equation for c does not depend on the auxiliary points. Therefore any equational computation to get extensions will not be affected by the auxiliary points.

2. Let us now take a loop involving joint attacks. Suppose we have 3 items, a , b and c and enough money to buy only two. Thus buying any two items attacks jointly the buying the third item. We get the loop of Figure 21.

The representation of this figure using the auxiliary point into an ordinary argumentation network is presented in Figure 22.

In this figure, the set $\{a, b, c\}$ is conflict free in the expanded network with the auxiliary points but it is not so in the original network, contrary to intuition. So CF2 messes up the concept of joint attack.

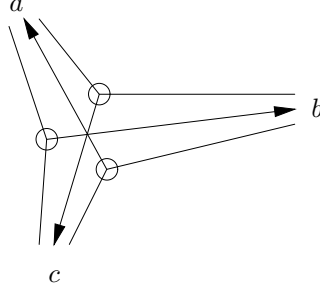


Figure 21.

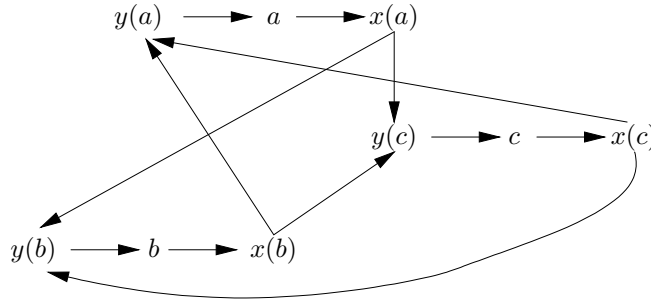


Figure 22.

In comparison, the LB semantics is not affected by the auxiliary points as we have already seen from the equations. Not being affected means that if we start with a network (S, R) and move for whatever reason to an expanded network (S', R') , containing additional auxiliary points, then any extension E' obtained traditionally for (S', R') will endow a correct and acceptable extension $E = E' \cap S$ of (S, R) , and furthermore all such correct extensions E are so obtained.

Let us find a loop-buster for Figure 22. Note that this system is similar to Figure 8, being a 9-point loop.

A computational loop-buster would be, for example, $\{y(a), y(b)\}$.

This gives $a = b = 1$ and $c = 0$. $\{y(a), y(b), y(c)\}$ is not minimal so we cannot get the extension $\{a, b, c\}$.

It is clear that our LB machinery works correctly here.

6 Discussion and conclusion

Let us summarise the progression of logical and conceptual steps involved in this paper:

1. We saw that the equational approach allows us to associate in a one to one manner a system of equations with each argumentation network.

2. Finding a solution to the equations corresponds to finding an extension for the argumentation network.
3. When a system of equation is difficult to handle, there is a well known and much used methodology in the equational area of perturbing the equations to make it them more manageable, in a way compatible with one's application area.
4. In the argumentation area odd and even loops are a bit of a problem
5. The counterpart of loops in the equational area are cycles of variable dependencies, a well understood phenomena, traditionally handled by iterative solutions.
6. We therefore suggest, inspired by equational thinking, the concept of loop-busting sets of variables which we perturb to be 0, and use them to modify the equations.
7. This gives rise to the LB semantics for argumentation, which contains the CF2 semantics as a special option, which we called LB2 (= CF2).
Note that LB2 is based on making arguments 0 , while CF2 is based on making arguments 1. So the agreement is non-trivial.
8. We considered other options such as LB1, LB3 and LB4. We showed that they agree with traditional semantics on even loops and still repair odd loops.
9. We compared the other LB semantics with CF2 and found them robust as far as conceptual changes such as joint attacks.
10. The LB1, LB3 and LB4 semantics can be done by loop busting using repeatedly and recursively one element at a time. The LB2 semantics (which simulates CF2 and uses maximal conflict free sets) requires the use of all points of the loop busting set to be made 0 simulataneously.

We now say a few comments, comparing this paper with [3]. The Shkop semantics introduced in [3], corresponds to LB4. In [3] argumentation loops are considered as created over time, where the arguments come as a result of agents creating situations by their temporal actions. This means that when an odd loop is created, we can identify one of the members of the loop as the temporally last argument which came into existence by some action and created the loop. The Shkop principle says that in this case, this argument is rejected (in the real world the action giving rise to it is annulled). This is mathematically equivalent to using this last argument as a loop buster and making it equal 0.

So the Shkop principle and the Shkop semantics of [3] corresponds to LB4 semantics where the loop-busters are chosen using temporal information.

Acknowledgements

I am grateful to Martin Caminada, Sarah Gaggl, and Stefan Woltran for helpful discussion.

Research done under ISF project: Integrating Logic and Network Reasoning.

BIBLIOGRAPHY

- [1] P. Baroni, M. Giacomin, and G. Guida. SCC-Recursiveness: A General Schema for Argumentation Semantics. *Artif. Intell.*, 168(1-2):162210, 2005.
- [2] S. A. Gaggl and S. Woltran. cf2 Semantics Revisited. In *Computational Models of Argument: Proceedings of COMMA 2010*, Desenzano del Garda, Italy, September 8-10, 2010, P. Baroni, F. Cerutti, M. Giacomin and G. Simari, eds., pp. 243–254. Volume 216 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2010.
- [3] M. Abraham, D. Gabbay and U. Schild. The handling of loops in Talmudic Logic, with application to odd and even loops in argumentation, Expanded Version. An earlier version appeared in *Proceedings of Howard 60*, Dec 2011, Editors D. Rydeheard, A. Voronkov and M. Korovina, pp 1-25.
- [4] D. Gabbay. An Equational Approach to Argumentation Networks, Feb 2011, 104 pp, to appear in *Argumentation and Computation*.
- [5] H. Prakken and G. A. W. Vreeswijk. Logics for defeasible argumentation. In D. M. Gabbay, F. Guenther (Eds.), *Handbook of Philosophical Logic*, Second Edition, Kluwer Academic Publishers, Dordrecht, 2001.
- [6] P. Baroni and M. Giacomin. Solving semantic problems with odd-length cycles in argumentation. In *Proceedings of the 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2003)*, pp. 440–451. LNAI 2711, Springer-Verlag, Aalborg, Denmark, 2003.
- [7] Pietro Baroni, Martin Caminada and Massimiliano Giacomin. An introduction to argumentation semantics, *The Knowledge Engineering Review* (November 2011), 26 (4), pg. 365-410
- [8] Sarah Alice Gaggl and Stefan Woltran. The cf2 Argumentation Semantics Revisited. To appear in *Journal of Logic and Computation*, 2012.
- [9] D. Gabbay. Fibring Argumentation Frames, *Studia Logica*, 93(2-3): 231–295,2009,.