Theorems and Proofs

The following algorithm, from "Proof Theories and Algorithms for Abstract Argumentation Frameworks", by Sanjay Modgil and Martin Caminada, produces the (unique) *grounded* extension of (A,R). It is known to be the *intersection* of all complete extensions, so it should not be used to find complete extensions, of which there can be many.

Initialize IN(L0)=OUT(L0)=UNDEC(L0) = {}
Repeat until no change {
  IN(Li+1) = IN(Li) $\cup$ {x | x is not labelled in Li, and
    for all y : if yRx then y $\in$ OUT(Li) }
  OUT(Li+1) = OUT(Li) $\cup$ {x | x is not labelled in Li, and
    for some y : yRx and y $\in$ IN(Li+1) }
}
UNDEC := Arg − IN − OUT

Theorem 1. *Acyclic DAFs have a unique complete extension, which is grounded (labeling everything with IN or OUT, but not UNDEC).*

Proof. We use the grounded labeling algorithm, which starts with everything not attacked being IN.

Proof induction on the number n of nodes A in (A,R).

Base case: n=1. Acyclicity implies R={}; so everything is labelled IN. QED

Indn step: n+1. Since R is acyclic and non-empty, it has at least one greatest/top element z, which does not attack anything. So $(A - \{z\}, R - \{(*, z)\})$ is also acyclic and by induction hypothesis has unique extension S′ labeling nodes IN′, and OUT′=(A-{z})-S′.
Now add back attacks {(b_1,z),...,(b_n,z)}
* if some b_k was in S′, then z -must- be labelled OUT, so IN remains unchanged, and OUT=OUT′ $\cup$ {z}
* if every b_i was OUT in S′, then IN=IN′ $\cup$ {z}, and OUT=OUT′
* Because S′ labelled everything IN or OUT, there is no other choice in S
QED

Definition 1. *An* Extended RefTree/Forest (XRF) *is obtained from an ordinary RG that is a tree by eliminating some edges.*

An XRF is a forest with goal, defect and refinement roots, where

- a goal node has 0 or more defect children, and 0 or 1 refinement parent;
- a defect node has 0 or more refinement children, and 0 or 1 goal parent it attacks;
- a refinement node has 1 or more goal children, and 0 or 1 addressed defect parent.

ASPIC+ translation:
goal g $\rightsquigarrow$ argument g

+ defeasible assumption g
defect d $\rightsquigarrow$ argument d
+ defeasible assumption d
+ contrary (d,g) if d targets g
refinement r $\rightsquigarrow$ argument r
+ rule (g1 $\wedge$ ... $\wedge$ gn $\Rightarrow$ r)
+ contrary (r,d) if r addresses d

/* in the earlier notation, {g,d,r} were $\{id_g, id_d, id_r\}$ */

As a result, in the translation of a XRF, for a defect d, either

* d is stand-alone,
* d -REBUTS- one (top-level) g, in InitR, or
* d -REBUTS- one (subgoal) g, and -UNDERMINES- one refinement Rr (when g in subgoals(r))

NOTATION: **Rr** refers to the proof tree with root r and children g1,... , which becomes a DAF node when ASPIC+ is translated to DAF

In the DAF generated from the above ASPIC+ translation of an XRF:

* goal arg g does not attack anything; it is attacked by zero or more d;
* defect arg d attacks zero or one top level goal arg g -or- both the goal g and refinement Rr (when g is in subgoals(r)). Since goal g cannot be reused in multiple refinements, d attacks at most one refinement Rr. (Note: A defect d can be attacked by more than one refinement arg Rr)
* refinement arg Rr attacks at most one d; it is (undermine) attacked by 0 or more defects d

Horn Theory HT translation:

goal g with defect children d_1,... d_n $\rightsquigarrow$
g `:-` hd_1,... hd_n; (the name hd is short for "handled d")
(if $n = 0$, the rule is g `:- true`)

refinement r with subgoal children g1,... $\rightsquigarrow$
r `:-` g_1,... g_n;

refinement r addresses defect $\rightsquigarrow$
hd `:-` r;

NOTATION: **DAF(RG)** $\models$ **x:IN/x:OUT** iff arg x is in/not in the unique grounded extension of DAF(RG).

Lemma 1. *i. DAF(RG)* $\models$*g:IN iff HT(RG)* $\models$*g*
ii. DAF(RG) $\models$*d:IN iff HT(RG)* $\not\models$*hd*
(Equivalently, DAF(RG) $\models$*d:OUT iff HT* $\models$*hd, because DAF is ground.)*
iii. DAF(RG) $\models$*Rr:IN iff HT(RG)* $\models$*r*

Proof.

In an XRF, we call a *"top contra edge (x,y)"* one of the following:

[1] d0 contra InitR -root- goal g0 (g0 not subgoal of a refinement)

[2] d0 contra subgoal g0 of refinement -root- node r0
[3] r0 contra defect -root- node d0

Given a regular Refinement Graph T that is a forest with at least one contra edge, let elim(T) be an XRF where a top contra edge in T is removed. [If T has a contra edge then it must have a top contra edge, because one can follow the path up from the non-top contra edge towards a root.] (Use some ordering on edges to make this deterministic.) If T has n contra edges, let $T_n \ := \ T, \ T_{i-1} \ := \ elim(T_i)$. Note that InitR is defined wrt to T, not T_i.

We then re-introduce contra edges one by one, by induction on the index n of T_n. /* Should probably be inside an induction on # of nodes.*/

BASE CASE: n=0;
So the XRF T0 has as roots
- all goals g (un-targeted by any defects)
- all refinements r, with subgoal children g1,...
- all defects d

In HT, there are
- goal rules g `:-` true, for all g
- refinement rules r `:-` g1,...
- no hd `:-` r rules

So HT $\models$ x iff x=g or x=r, and HT $\not\models$ hd for all d.

In DAF, there are no Attacks, so everything is IN.

So
(i) DAF(T0) $\models$ g:IN , HT(T0) $\models$ g /* for all g */
(ii) DAF(T0) $\models$ d:IN for all d , HT(T0) $\not\models$ hd for any hd
(iii) DAF(T0) $\models$ Rr for all Rr, since all args are IN
HT(T0) $\models$ r for all r, because all g `:-` true rules fire,
and then all r `:-` __ rules fire

QED base case

INDN HYPOTHESIS: Assume lemma is true for XRF T_n (call it $\mathbf{T'}$, with Horn theory $\mathbf{HT'}$, and Dung framework $\mathbf{DAF'}$ generated from ASPIC+)

INDN STEP: RTP lemma for XRF T_{n+1} (call it **T**, **HT**, and **DAF** respectively)
CASES for edge e removed from T:
[1] d0 contra InitR root goal g0 (not subgoal of a refinement)
[2] d0 contra subgoal g0 of refinement root node r0
[3] r0 contra defect root node d0

[[case 1]] d0 contra InitR root goal g0
and g0 does not appear as subgoal for any refinement.

HT = HT′ – {g0 :- restHd} ∪ {g0 :- hd0,restHd}
So $Consequences(HT) \subseteq Consequences(\mathrm{HT}')$ because change only makes one rule harder to fire. The inequality holds iff
(ht1.1) HT′ $\models$ g0; but HT$\not\models$ g0 because
(ht1.2) HT′$\not\models$ hd0
in which case $Consequences(HT) = Consequences(\mathrm{HT}') - \{g0\}$
because g0 did not appear in other rules

DAF = DAF′ ∪ {d0 attacks g0}
so $extension(DAF) \subseteq extension(\mathrm{DAF}')$ because an attack may affect g0, but g0 does not attack anything else (it is not a subgoal).
The inequality holds iff
(daf1.1) DAF′ $\models$ g0:IN; but DAF $\models$ g0:OUT because
(daf1.2) DAF′ $\models$ d0:IN
in which case $Extension(DAF) = Extension(\mathrm{DAF}') - \{g0\}$
because g0 is not a subgoal, so it does not attack anything else.

But
by induction hypothesis (i): (ht1.1) iff (daf1.1)
by induction hypothesis (ii): (ht1.2) iff (daf1.2)
QED [1]

[[case 2]] d0 contra subgoal g0 of refinement root node r0

HT = HT′ – { g0 :- restHd } + {g0 :- hd0,restHd }
so $Consequences(HT) \subseteq Consequences(HT')$ because change makes one rule harder to fire. The inequality holds iff
(ht2.1) HT′ $\models$ g0 (hence HT′ $\models$ restHd) but HT $\not\models$ g0,
(ht2.2) so HT′ $\not\models$ hd0 since still HT′ $\models$ restHd
in which case
$Consequences(HT) = Consequences(\mathrm{HT}') - \{g0\}$
and *possibly*
$- \{r0\}$ if
(ht2.3) HT′ $\models$ r0, because now HT $\not\models$ r0
[g0 appears in rhs of rule r0 :- g0,...
and there was no _ :- r0,... rule to propagate
this, as r0 is a root]

DAF = DAF′ ∪ {d0 attacks g0, d0 attacks Rr0} so $extension(DAF)$ $\subseteq$ $extension$(DAF′) because a new attack may only affect g0 and Rr0 (these do not attack anything else, as roots). The inequality holds iff

(daf2.1) DAF′ $\models$ g0:IN and DAF $\models$ g0:OUT) because

(daf2.2) DAF′ $\models$ d0:IN

in which case

$Extension(DAF)$ = $Extension$(DAF′) − $\{g0\}$

and *possibly*

− $\{Rr0\}$ if

(daf2.3) DAF′ $\models$ Rr0:IN, bec. now DAF $\models$ Rr0:OUT

But

by induction hypothesis (i): HT′ $\models$ g0 iff DAF′ $\models$ g0:IN

by induction hypothesis (ii): HT′ $\not\models$ hd0 iff DAF′ $\models$ d0:IN

by induction hypothesis (iii): HT′ $\models$ r0 iff DAF′ $\models$ Rr0:IN

QED [2]

[[case 3]] r0 contra defect root node d0

HT = HT′ ∪ {hd0 :- r0}

$Consequences$(HT′) $\subseteq$ $Consequences(HT)$, since a rule is added. The inequality holds iff

(ht3.1) HT′ $\models$ r0, in which case HT $\models$ hd0, but

(ht3.2) HT′ $\not\models$ hd0 (there could have been other hd0 :- r_i, but HT ′ $\not\models$ r_i)

in which case $Consequences(HT)$ = $Consequences$(HT′) ∪ $\{hd0\}$

because, as a root, hd0 does not appear in any rule _ :- hd0,...

DAF = DAF′ ∪ {Rr0 attacks d0}

so $extension(DAF)$ $\subseteq$ $extension$(DAF′) because an attack may affect d0, but d0, as a root, does not affect anything else.

The inequality holds iff

(daf3.1) DAF′ $\models$ Rr0:IN, in which case DAF $\models$ d0:OUT, but

(daf3.2) DAF′ $\models$ d0:IN

in which case $Extension(DAF)$ = $Extension$(DAF′) − $\{d0\}$

since d0, as a root, does not attack anything else.

But by induction hypothesis (iii): (ht3.1) iff (daf3.1)

by induction hypothesis (ii): (ht3.2) iff (daf3.2)

QED [3]

The following theorem is a consequence of Lemma 1, item (i).

Theorem 2. *Given a requirements graph RG that is a tree/forest, the set of goals derivable from HT(RG) is identical to the set of goals appearing in the unique complete and grounded extension of ASPICT(RG).*