ARGUMENTATIE VIA GESIMULEERDE DIALOGEN

Bachelorproject

Bas Hickendorff , s1533355 , s.r.hickendorff@ai.rug.nl , begeleider: Bart Verheij

Samenvatting: Bij argumentatie worden argumenten en tegenargumenten gebruikt om te bepalen welke beweringen wel en welke beweringen niet volgen uit andere beweringen. Om dit geautomatiseerd te kunnen doen is het nodig argumentatie te formaliseren. In eerder onderzoek is een methode voor het formaliseren van argumentatie voorgesteld door Dung. In zijn onderzoek is echter weinig aandacht besteed aan discussie tussen meerdere agenten. In dit onderzoek wordt gekeken hoe de theorie van Dung toegepast kan worden op een dialoog tussen twee agenten, die elk hun eigen kennis van de wereld hebben. Hiervoor wordt een strategie ontwikkeld aan de hand van het 'proofs and refutations' algoritme van Verheij. Via dit algoritme is het mogelijk de status van een argument te controleren zonder de status van alle argumenten na te hoeven gaan. Het blijkt dat het mogelijk is de strategie te koppelen met de theorie van Dung.

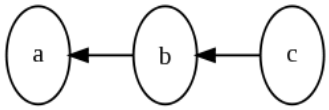
1. Inleiding

Bij argumentatie worden argumenten en tegenargumenten gebruikt om te bepalen welke beweringen wel en welke beweringen niet volgen uit andere beweringen. Dat is een belangrijke vaardigheid voor intelligente systemen. Deze studie kijkt naar een manier om argumentatie te onderzoeken, namelijk via dialogen.

Wanneer argument B er voor zorgt dat argument A niet meer kan kloppen, noemen we dat een aanval van argument B op argument A. Als het dan zeker is dat B correct is, is ook zeker dat A dat niet is. Bijvoorbeeld:

A: verdachte heeft niets gestolen
B: getuige zegt dat verdachte iets gestolen heeft
C: getuige liegt

Als alleen A bekeken wordt, is de conclusie dat de verdachte niets gestolen heeft. Als B dan toegevoegd wordt, moet die conclusie ingetrokken worden, en is de conclusie dat de verdachte wel iets gestolen heeft. Wanneer dan ook C toegevoegd wordt verandert de conclusie weer in niets gestolen. De conclusie die getrokken wordt kan dus verworpen worden wanneer er meer informatie wordt toegevoegd. Dit heet *defeasible argumentation*. Buiten het gebied van de argumentatie wordt dit niet-monotone logica genoemd. Het bovenstaande voorbeeld kan weergeven worden in een netwerk, zie figuur 1.



Figuur 1

Een geautomatiseerde vorm van argumentatie kan ondersteuning bieden aan het nemen van een beslissing, zoals een uitspraak in een rechtszaak en heeft veel toepassingen in multi-agent systemen. Zie [1] voor een overzicht van verschillende toepassingen en technieken. Om argumentatie te automatiseren is het nodig de theorieën hierover eerst te formaliseren. Een manier om dit te doen is het weergeven van de argumenten en hun relaties als een netwerk.

Dung [2] heeft een aantal begrippen gedefiniëerd die veel gebruikt worden in het onderzoek naar argumentatie. De begrippen die van belang zijn voor dit onderzoek volgen nu:

Een *argumentation framework* (AF) is als volgt opgebouwd: <Args , Attacks> . In Args bevinden zich alle argumenten, en Attacks zijn de aanvalsrelaties die er tussen die argumenten bestaan. Het is ook mogelijk om meer soorten relaties tussen argumenten te definiëren, zoals ondersteuning, maar dat wordt hier buiten beschouwing gelaten.

Een argument A is *acceptable* ten opzichte van een set argumenten S, wanneer voor alle argumenten die A aanvallen er een aanvaller in S te vinden is. Een set S is *admissible* wanneer elk argument in S acceptable is ten opzichte van. S.

Een *extensie* zou je kunnen omschrijven als het resultaat van een set argumenten. Het is een subset van de argumenten in het AF, met bepaalde eigenschappen. Er zijn verschillende soorten extensies gedefinieerd. Twee belangrijke typen extensies voor dit onderzoek zijn de *preferred* en de *stabiele extensie*. De preferred extensie is de grootst mogelijke admissable set in het AF, en de stabiele extensie is een set argumenten waarvoor geldt dat de argumenten in de extensie alle argumenten die niet in de extensie zitten aanvallen.

Aangezien de extensie fungeert als het resultaat van een eventuele discussie over de argumenten, mag de extensie niet in tegenspraak zijn met zichzelf. Bovenstaande extensies moeten daarom *conflict-free* zijn, dat wil zeggen, geen enkel argument in de extensie mag een element in de extensie zelf aanvallen.

De methode van Dung heeft geen directe manier om aan te geven dat een bepaald argument ondersteund wordt door een ander. Dat kan alleen door een aanvaller van dat argument aan te vallen. Er zijn ook andere methoden ontwikkeld die deze methode wel hebben, een voorbeeld daarvan is DefLog [3]. In deze methode is het ook mogelijk aanvalsrelaties zelf aan te vallen, zodat de structuur van het netwerk kan worden bediscussieerd. Dit is niet mogelijk in het protocol dat in dit onderzoek voor de dialogen gebruikt wordt.

In dit onderzoek wordt geprobeerd een extensie te bereiken via een dialoog. Een dialoog bestaat uit een aantal zetten tussen twee of meer agenten. Deze zetten moeten voldoen aan een bepaald protocol, waarin gedefinieerd is welke zetten op welk moment gedaan mogen worden, op welke manier de zet uitgevoerd wordt en wanneer de dialoog stopt. De agenten kunnen vervolgens een eigen strategie hebben die aangeeft welke van de (volgens het protocol legale) zetten uitgevoerd moet worden.

Vreeswijk en Prakken [4] geven een manier om via een dialoog *admissable* sets te produceren en daarmee te controleren of een bepaald argument onderdeel is van een preferred extensie. Beide spelers geven hier beurtelings een argument, dat een argument van de ander aanvalt (en daarmee mogelijk een eerder eigen argument verdedigt). Op het moment dat een van beide dat niet meer kan, kan er wel of geen set gemaakt worden (als de aanvaller van het oorspronkelijke argument niet meer kan, kan er een admissible set gemaakt worden van de argumenten die door de verdediger zijn genoemd, als de aanvaller van het oorspronkelijke argument niet meer kan, is dat niet mogelijk). Door een dergelijke methode wordt duidelijk wat de relatie tussen een bepaalde extensie en een manier van discussiëren is, aangezien de extensie logisch volgt uit de regels die er voor de dialoog zijn opgesteld.

In het onderzoek van Dung worden dialogen buiten beschouwing gelaten. Daarom is het doel van dit onderzoek het ontwerpen van een aantal

strategieën, die wanneer ze door twee agenten worden toegepast, resulteren in de extensies die Dung gedefinieerd heeft. In dit onderzoek wordt gekozen voor de stabiele extensie omdat deze de meest complete vorm van een uitkomst zonder conflicten is, alle argumenten moeten bekend zijn, en alles wat niet bewezen kan worden, moet aangevallen worden. De strategieën kunnen gebruik maken van het *proofs en refutations* algoritme van Verheij [5]. Dit algoritme wordt in de volgende paragraaf verder besproken. Beide agenten zullen een gedeelte van de argumenten en aanvalsrelaties van het totale netwerk bezitten en samen kunnen ze dan proberen een volledige extensie te creëren.

De onderzoeksvraag is: *Welke strategieën zorgen ervoor dat agenten in een argumentatieve dialoog gedeelde interpretaties van hun kennis van de wereld creëren die voldoen aan Dung's semantieken?*

Eerst zal het 'proofs en refutations' algoritme besproken worden, daarna twee mogelijke strategieën, en tot slot bekijken we of deze strategieën er inderdaad voor zorgen dat de agenten op een gezamelijke interpretatie van de argumenten uitkomen.

2. Het proofs and refutations algoritme

Het proofs and refutations algoritme [5] heeft als doel een set argumenten te vinden die het argument waarop het algoritme wordt toegepast ondersteunen of juist aanvallen. Hieronder volgt een korte beschrijving van het algoritme.

Het algoritme maakt gebruik van labelings, die aan één of meer argumenten een label toekent. De labeling die in dit algoritme gebruikt wordt, heeft de vorm <J,D> . In J zitten alle argumenten waarvan aangenomen wordt dat ze verdedigbaar zijn. In D zitten alle argumenten waarvan aangenomen wordt dat ze weerlegd kunnen worden. Argumenten kunnen niet in beide labelings zitten, aangezien dat een contradictie zou opleveren. Hieronder volgt een beschrijving van de functies van het algoritme. Het algoritme wordt gestart via de PartialProofOptions en/of de PartialRefutationOptions functie.

extendByAttack(J,D):
Breidt labeling D uit met alle aanvallers van de argumenten in labeling J

extendByDefense(J,D):
Breidt labeling J zodanig uit dat er voor elk argument in labeling D een aanvaller in J zit, dat J zo klein mogelijk blijft en dat J *conflict-free* blijft.

PartialProofOptions $_0$ (Arg) := { Arg () }
PartialProofOptions $_{2n+1}$ (Arg) := ExtendByAttack(PartialProofOptions $_{2n}$ (Arg))
PartialProofOptions $_{2n+2}$ (Arg) := ExtendByDefense(PartialProofOptions $_{2n+1}$ (Arg))

PartialRefutationOptions $_0$ (Arg) := { (Arg) }
PartialRefutationOptions $_{2n+1}$ (Arg) := ExtendByAttack(PartialProofOptions $_{2n}$ (Arg))
PartialRefutationOptions $_{2n+2}$ (Arg) := ExtendByDefense(PartialProofOptions $_{2n+1}$ (Arg))

Twee functies staan centraal in het algoritme, ExtendByDefense en ExtendByAttack. ExendByAttack zorgt er voor dat alle aanvallers van alle argumenten, die in de justified labeling zitten, in de defeated labeling komen. ExtendByDefense zorgt dat voor elk argument, dat op dat moment defeated is, er minimaal één aanvaller in de justified labeling zit. Aangezien er meerdere argumenten kunnen zijn, die het defeated argument aanvallen, zijn er bij deze

functie verschillende uitkomsten mogelijk. Deze moeten allemaal uitgerekend worden, aangezien van te voren nog niet te zien is welke van deze argumenten uiteindelijke onderdeel zijn van een proof of een refutation. Door deze eigenschap zorgt extendByDefense ervoor dat er uiteindelijk meerdere proofs en refutations voor een argument gevonden kunnen worden.

Door deze twee functies af te wisselen krijgen alle argumenten, die het argument waar je mee begint direct of indirect aanvallen, een labeling. Als er tijdens dit proces aanvallende argumenten (ten opzichte van het argument waar mee begonnen werd) gevonden worden waar geen verdediging voor is, is er geen bewijs voor dat argument. Omgekeerd, als er geen aanvallende argumenten zijn voor van een argument dat defeated moet worden, is er geen verwerping van dat argument.

Het is mogelijk dat er zowel een proof als een refutation bestaat (dit is bijvoorbeeld het geval als A B aanvalt, en B A aanvalt). Het niet hebben van een proof is daarom iets anders dan het hebben van een refutation voor een argument, en omgekeerd.

Wanneer niet alle aanvallers van een argument verworpen kunnen worden, bestaat er geen proof van dat argument. Wanneer niet voor minstens 1 aanvaller van een argument een proof gevonden kan worden, bestaat er geen refutation voor dat argument.

3. Beschrijving strategie 1

Zoals in de inleiding beschreven gaat het erom dat twee agenten samen een stabiele extensie maken. Er is gekozen voor twee agenten met een identieke strategie, maar een verschillend netwerk, eventueel wel met gedeeltelijke overlap. Het gaat om een coöperatieve dialoog, dus beide agenten hebben baat bij zo open mogelijk zijn over de kennis die ze hebben.

Wanneer we het realisme van een dialoog buiten beschouwing laten, is de meest eenvoudige oplossing dat beide agenten direct al hun argumenten opnoemen. Dan zijn alle argumenten bij beide agenten bekend en is er geen onzekerheid meer over de status van die argumenten. Daar wordt hier niet voor gekozen, omdat geprobeerd wordt het proces van het ontdekken van bepaalde argumenten weer te geven. Daarnaast staat het achter elkaar opnoemen van alle argumenten heel ver van de realiteit af. Er wordt daarom gekozen voor een strategie die uitgaat van de aanvallen waar nog geen verdediging voor is, en alleen daarop reageert.

Het protocol dat voor de dialogen gebruikt wordt staat de volgende zetten toe: A , xA , ¬ A , ¬ xA , A: B , xA: B , ¬ A: B , ¬ xA: B . Waarbij A betekent dat de agent een proof heeft voor argument A, xA betekent dat de agent een refutation heeft voor argument A. en B weer een van de vier eerste zetten in de lijst mag zijn. ¬ is de logische negatie. Verder mogen de agenten geen zetten herhalen.

De strategie ziet er als volgt uit:

In de strategie worden de acties beschreven die 1 agent uitvoert. Deze agent heet agent X, de agent waar de discussie mee gevoerd wordt heet agent Y.

Strategie 1:

stap 1: Als er al iets gezegd is, het laatste argument dat genoemd is door agent Y aan het netwerk van agent X toevoegen als dat argument nog niet in het netwerk van agent X

zit. De labeling die agent Y aan dat argument toegekend heeft opslaan.

stap 2: De proofs en refutations uitrekenen van alle argumenten die agent X kent, en voor elk argument de toegekende labeling opslaan.

stap 3: Voor elk argument A waar agent X een proof voor heeft, A aan lijst M toevoegen. Wanneer agent X een refutation voor A heeft, xA aan M toevoegen. Dan, wanneer de agent X geen proof voor A heeft ¬A aan M toevoegen, en wanneer X geen refutation voor A heeft ¬xA aan M toevoegen. Al deze zetten worden alleen toegevoegd wanneer ze nog niet de dialoog zijn voorgekomen.

stap 4: De labeling van elk argument dat agent X heeft vergelijken met de labeling waarvan agent X weet dat die door agent Y aan dat argument toegekend wordt. Wanneer de labeling van een argument verschilt, dat argument toevoegen aan M, samen met het eerste argument uit alle gevonden proofs of refutations voor dat argument dat nog niet genoemd is, dit is de motivatie voor de zet.

stap 5: Elk bij agent X bekend argument langslopen en controleren wat de laatst door agent X genoemde labeling van dat argument is. Wanneer die labeling verschilt van de huidige labeling van het argument bij agent X, dat argument met die labeling toevoegen aan M.

stap 6: Willekeurig een zet uit M kiezen om uit te voeren. Daarna M leegmaken. Wanneer beide agenten achter elkaar geen zet hebben is de dialoog afgelopen.

Een voorbeeld van een dialoog met agenten met deze strategie is te vinden in bijlage 1.

Het is met deze strategie niet mogelijk een dialoog te krijgen met meer zetten dan vier keer het aantal argumenten. Dat komt omdat zetten niet herhaald mogen worden. Er zijn dus vier varianten mogelijk bij elk argument: proof, refutation, geen proof en geen refutation. Voor een argument kunnen deze varianten uiteraard niet allemaal tegelijk waar zijn, maar door het geleidelijk beschikbaar komen van nieuwe informatie kunnen alle vier de varianten wel in de dialoog voorkomen.q

Om zeker te weten dat beide agenten van alle argumenten op de hoogte zijn, is het nodig alle argumenten twee keer te behandelen. Dit omdat pas na de eerste keer alle argumenten bekend zijn. Bij de tweede keer is het dus gegarandeerd dat het hele netwerk bekend is. Hierdoor is het zo dat na de dialoog beide agenten een identiek netwerk van argumenten hebben, en daardoor ook identieke proofs en refutations voor elk argument. Van deze losse proofs en refutations moet nu nagegaan worden of er een stabiele extensie te maken is.

4. Het combineren van de proofs en refutations

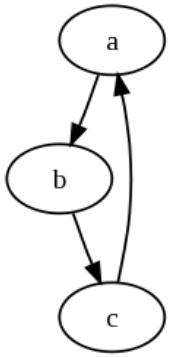
Het proofs en refutation algoritme levert informatie op over de bewijsbaarheid van één argument, maar het doel van de te ontwikkelen strategie was het creëren van een stabiele extensie. Om een stabiele extensie te kunnen maken, is het nodig de labeling van alle argumenten te weten. Dit kan door te proberen de verschillende proofs en refutations die er per argument zijn aan elkaar te koppelen.

Voor elk argument worden alle mogelijke proofs en refutations bepaald, en daarna wordt bekeken of er een combinatie van deze proofs en refutations is, waarbij ze allemaal het argument dezelfde status geven. Het feit dat er voor elk argument één of meerdere correcte proofs of

refutations zijn, wil nog niet zeggen dat zo'n combinatie ook bestaat. Dit is alleen uit te vinden door alle combinaties te proberen, net zo lang tot er een extensie gevonden is.

Het is mogelijk dat er voor een argument zowel een proof als een refutation gevonden is (dit is mogelijk wanneer er een cirkel in het netwerk zit). Dat wordt dialectische ambiguïteit genoemd. Voor het maken van een extensie moet er hieruit één optie gekozen worden waarbij geen conflict ontstaat. Het omgekeerde, wanneer er geen proof en geen refutation gevonden wordt, wordt dialectisch niet-interpreteerbaar genoemd. Wanneer een of meerdere van dergelijke argumenten in het netwerk voorkomen, is het niet mogelijk om een stabiele extensie te maken, aangezien voor de stabiele extensie voor elk argument een beslissing moet worden genomen over de status die dat argument heeft.

Wanneer twee agenten over alle argumenten proofs en refutations hebben uitgewisseld, is het dus nodig daar een selectie uit te maken. Dat kan op een vergelijkbare manier als het uitwisselen van de argumenten. De agenten hebben niet noodzakelijk dezelfde proofs en refutations beschikbaar, dus wanneer een agent een proof wil inbrengen dat de ander nog niet kent, kan het nodig zijn terug te gaan naar de fase van het uitwisselen van die argumenten. Wanneer dit niet gebeurt, is het meestal niet meer mogelijk om op een stabiele extensie uit te komen, aangezien de andere agent dan vrijwel zeker onjuiste waarden aan bepaalde argumenten heeft toegekend.



Figuur 2

Helaas bestaat er niet altijd een combinatie van proofs en refutations die geen conflicten oplevert. Er is bijvoorbeeld geen oplossing bij netwerken met een cirkel met een oneven aantal argumenten (zie figuur 2). Het construeren van een stabiele extensie lukt dan niet, aangezien het niet mogelijk is alle argumenten, die niet in de extensie zitten, aan te vallen en tegelijkertijd binnen de extensie geen conflicten te introduceren. Omgekeerd is het ook mogelijk dat er twee stabiele extensies zijn, alle argumenten die dan bij de eerste justified zijn, zijn bij de tweede defeated, en omgekeerd. Dat komt voor bij cirkels met een even aantal argumenten.

In de opzet van het onderzoek was het de bedoeling dat de agenten, na over de proofs en refutations gediscussieerd te hebben, een discussie zouden voeren over welke proofs en refutations samengevoegd moeten worden tot een extensie. Het blijkt echter dat dit geen meerwaarde heeft, aangezien, zoals bij de eerste strategie bleek, de agenten na de discussie over proofs en refutations gegarandeerd zelf alle informatie al hebben. Informatie uitwisselen in een dialoog is dan dus niet nodig, de agenten kunnen zelf de combinaties van de proofs en refutations proberen te maken.

5. Beschrijving strategie 2

In de tweede strategie wordt, in plaats van met twee agenten het hele netwerk van argumenten in kaart te brengen, een discussie gevoerd over één argument. De stabiele extensie is dan niet bruikbaar, omdat daarvoor altijd alle argumenten een labeling moeten krijgen. Er wordt daarom gekozen voor de preferred extensie. De agenten gaan proberen een preferred extensie bij het bediscussieerde argument te vinden. We gaan er hierbij vanuit dat de agenten voor het bediscussieerde argument het nog niet eens zijn, aangezien, wanneer ze het wel eens zijn die dialoog direct stopt. Het protocol dat gebruikt wordt bij een dialoog met deze strategie is gelijk aan het protocol bij strategie 1.

Elke beurt beginnen beide agenten bij stap 1, er staat één argument centraal, dat is het bediscussieerde argument.

In de strategie worden de acties beschreven die 1 agent uitvoert. Deze agent heet agent X, de agent waar de discussie mee gevoerd wordt heet agent Y.

Strategie 2:

Stap 1: Als er nog niets gezegd is, de labeling noemen die het bediscussieerde argument bij agent X heeft. Daarna is deze beurt afgelopen voor agent X. Als er al wel iets gezegd is, ga naar stap 2.

Stap 2: Het argument dat als laatst genoemd is door agent Y aan het netwerk van agent X toevoegen als die er nog niet inzat. De labeling die agent Y aan dat argument toekent opslaan.

Stap 3: Controleren of de labeling van het bediscussieerde argument bij agent X verschilt met de labeling die agent Y aan dat argument toegekend heeft. Als dat niet zo is, zeg dat, en stop de dialoog.

stap 4: De proofs en refutations uitrekenen van alle argumenten die agent X kent, en voor elk argument de toegekende labeling opslaan.

stap 5: De labeling van alle argumenten die bij agent X bekend zijn vergelijken met de labeling waarvan agent X weet dat die door agent Y aan dat argument toegekend wordt. Wanneer deze labeling bij een argument verschilt, dat argument aan M toevoegen, samen met het eerste argument uit alle proofs of refutations voor dat argument dat nog niet genoemd is, dit is de motivatie voor de zet.

stap 6: Willekeurig 1 argument uit lijst M kiezen als zet. Daarna M leegmaken.

Een voordeel van deze strategie is dat alleen de argumenten die relevant zijn voor het centraal staande argument besproken worden. Deze strategie garandeert dat argumenten die deel uit maken van een proof of van een refutation van een ander besproken argument dan het bediscussieerde argument, ook deel uit maken van de proof dan wel refutation van het bediscussieerde argument.

Wanneer de dialoog stopt hebben beide agenten een preferred extensie met daarin het bediscussieerde argument. Hoewel de agenten het over dit argument eens zijn, kunnen de redenen hiervoor verschillen, en de extensies dus ook. De preferred extensie van de ene agent kan argumenten bevatten die de andere agent niet kent.

Deze strategie heeft een aantal eigenschappen die lijken op het proofs en refutations algoritme (alhoewel dat algoritme zelf niets met dialogen te maken heeft). Het gaat er eigenlijk om dat de agenten allebei een proof vinden, of allebei een

refutation voor het besproken argument. Eerder is al besproken dat er voor een argument meerdere proofs kunnen zijn, en in dit geval zou de ene proof in het netwerk van de ene agent kunnen kloppen, terwijl in het netwerk van de andere agent juist de andere proof te vinden is.

6. Discussie

Dit onderzoek ging vooral om het toepassen van proofs en refutations op dialogen. In het oorspronkelijke onderzoek van Dung [2] worden dialogen niet besproken. Daarna is er wel enig onderzoek gedaan naar dialogen, maar de vernieuwing van dit onderzoek is het gebruik van het 'proofs and refutations' algoritme. [5].

Een ander verschil met ander onderzoek is dat het in dit onderzoek gaat om een coöperatieve dialoog, in tegenstelling tot bijvoorbeeld [4], waar de dialoog eigenlijk een spel is dat gewonnen kan worden. Bij de 2e strategie is dit verschil echter niet groot. We zouden makkelijk de agent die zijn oorspronkelijke mening over het bediscussieerde argument moet bijstellen als verliezer kunnen beschouwen.

Dialogen zijn een goede manier om argumentatie te onderzoeken, aangezien argumentatie veel gebruikt wordt in de interactie tussen mensen. Een gesimuleerde dialoog is dan een realistischer benadering dan alleen het onderzoeken van een netwerk van argumenten. Hoewel een formeel model altijd enigszins abstract zal zijn, zijn er een aantal aanpassingen en uitbreidingen denkbaar die in dit onderzoek nog niet meegenomen zijn, maar de dialoog volgende uit het protocol en de strategie nog meer op een echte discussie zou laten lijken, zoals het niet vast leggen van de aanvalsrelaties, zodat die bediscussieerd kunnen worden, of het invoeren van andere relaties, zoals ondersteuning van argumenten.

7. Conclusie

De onderzoeksvraag was: *Welke strategieën zorgen ervoor dat agenten in een argumentatieve dialoog gedeelde interpretaties van hun kennis van de wereld creëren die voldoen aan Dung's semantieken?*

Dit kan beantwoord worden met strategie 1. Beide agenten kunnen van mening verschillen over de vraag of een argument een aanvaller heeft (dat zou een argument kunnen zijn dat een van beide nog niet heeft), maar als dat argument eenmaal genoemd is, zal de agent dat altijd accepteren, aangezien de agenten geen tegenstrijdige informatie kunnen hebben (de andere agent kan hoogstens weer een nieuwe aanvaller naar voren brengen). Deze beperking zorgt ervoor dat een aantal onderdelen van de protocollen en strategieën, die veel in de literatuur te vinden zijn, niet nodig zijn, zoals verschillende maten van zekerheid en verschillende manieren om een bepaald argument naar voren te brengen. Ook zorgt dit ervoor dat, als er een extensie bestaat in het gecombineerde netwerk van beide agenten, deze altijd gevonden zal worden (bij de eerste strategie). Hieruit is te concluderen dat met strategie 1 altijd voldaan wordt aan het doel van het onderzoek.

Strategie 2 komt iets dichter bij wat er bij een discussie vaak gebeurt. Wanneer er onenigheid is over een bepaalde stelling, zal men niet, zoals in strategie 1, alle beschikbare kennis bespreken, maar zich richten op de argumenten die bijdragen aan de conclusie die over die stelling.

Via strategie 2 kan dan ook makkelijk zichtbaar worden gemaakt welke argumenten belangrijk zijn voor de conclusie, iets wat bij strategie 1 minder makkelijk is.

literatuur

[1] Bench-Capon & Paul E. Dunne (2007) Argumentation in artificial intelligence *Artificial intelligence 171 (2007), p. 619 – 641*

[2] Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence 77, p. 321 - 357.*

[3] Verheij, B. (2003b) DefLog: on the logical interpretation of prima facie justified assumptions. *Journal of Logic and Computation 13(3), pp 319-346.*

[4] Vreeswijk & Prakken (2000) Credulous and sceptical argument games for preferred semantics *Lecture Notes in Computer Science (2000) vol. 1919/2008, p.239 - 253*

[5] Verheij, B. (2007). A Labeling Approach to the Computation of Credulous Acceptance in Argumentation. *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007 (ed. Manuela M. Veloso), p. 623 - 628.*

[6] P. McBurney, Parsons S., Wooldridge M. (2002) Desiderata for Agent Argumentation Protocols *Proceedings of the first international joint conference on Autonomous agents and multiagent systems, Session 4C (2002), p. 402 – 409*

[7] Caminada M. & Wu Y. (2009) An argument game for stable semantics *Logic journal of the IGPL [1367-0751] volume 17 issue 1, p. 77 - 90*

Bijlage 1:

Hieronder volgt een mogelijke uitkomst van een dialoog waarin de agenten strategie 1 gebruiken. Het complete netwerk was de volgende (eerste argument is het argument wat aangevallen wordt, de argumenten erachter zijn de aanvallers):

a b1 b2 b3
b1 c1
c1 d1
b2 c2 c3
c2 d2
d2 e1
b3 c4

Agent 1 kent aan het begin van de dialoog de argumenten a , d1, b2 en c2.
Agent 2 kent aan het begin van de dialoog de argumenten a, c1, b2, d2 en c4.

Aangezien de agenten in de laatste stap van de strategie een willekeurige zet uit de mogelijke zetten kiezen, ligt het verloop van de dialoog niet vast. Een mogelijk verloop is als volgt:

Agent 1	Agent 2
xc2	
	xb2
¬ xb2 : d2	
	c2 : e1
b3	
	a
¬ a : b1	

c3

xa

a : c4

d1

¬ xa : xd2