Collaborative Planning and Decision Support for Practical Reasoning in Decentralised Supply Chains

Naeem Khalid Janjua
School of Business
University of New South Wales, Canberra, Australia.
n.janjua@unsw.edu.au

Abstract

Current advanced systems for collaborative planning among supply chain trading partners are scalable in a deterministic environment, but their scalability does not translate for effective decision making when planning under severe uncertainties that arise because of imperfect knowledge and potentially contradictory preferences. This article addresses this gap and aims to explore the logical foundations and implementation of efficient and effective Collaborative Planning and Decision Support System (CP-DSS) for practical reasoning in decentralized supply chains. Using CP-DSS the planning information, which is incomplete and contradictory, can be shared, reasoned, integrated and visualized in a form which can be readily understood by all trading partners who can then challenge both the decision and the thinking that underpins the decision. Practical reasoning [1] underlines the need for informed and experienced judgment during planning and decision support and helps in building Systems of Engagement(SoE) [2] i.e., capturing and using peer-to-peer interaction information along with transactional data for decision support, which not only provides answers to the questions related to patterns of "*what is happening*" but also "*why it is happening and what is the rationale behind it*".

1 Introduction

Supply chain (SC) activities move the entire economy of the world and are one of the major contributors to nation's GDP [3]. The increase in outsourcing and the rise of digital technologies has led to the widespread adoption of e-business models and businesses are involved in collaboration and mergers with others on a global scale, competing as a SC rather than as individuals [4]. Due to the decentralized nature of SCs, the current SC trading partners are still far from structured collaborative planning and decision support in presence of information asymmetry. This means one SC trading partner might possess some important information (potentially incomplete and/or contradictory) that is neither available in the public domain nor verifiable by a third party. The inability to share and use such information in collaborative planning and decision support may result in suboptimal decisions.

Over the past decade, SC trading partners focused on building a *Systems of Record* (SoR) i.e. structured product data, orders and demand forecasts [5]. The decision support derived from the SoR can answer questions related to the pattern "*What is happening*" but provides no information to answer questions related to the pattern "*Why it is happening and what is the rationale behind it*". As a result, complex SC networks have a tendency to become vulnerable to uncertainties and operational risks [6]. Therefore, collaborative planning and decision support faces the challenge of aligning the activities of SC trading partners in a decentralized network which contributes to the value creation of the product or service for the customer in the presence of information asymmetry to overcome operational risks. This article aims to explore the logical foundations and implementation of efficient and effective CP-DSS for practical reasoning in decentralized supply chains. Although a large body of knowledge exists on collaborative planning and decision support, this approach of practical reasoning is novel due to the following two features:

1. *Incomplete and contradictory planning information is considered*: Developing and agreeing on plans that involve multiple players is not a trivial task, especially when there is a lack of information pertaining to the potential risk. In real life, when the underlying available information is incomplete and/or contradictory, people are reluctant to make decisions. As a result, they indulge themselves into the process of argumentation. Argumentation involves building arguments in favour and against a certain issue. Arguments are defeasible in nature i.e. additional information may invalidate what has been previously accepted as an argument. Propositions accepted on the basis of given arguments don't grow monotonically with the available information. Using this approach, not only can we reason and deliberate over the means by which choices are compared, we can critically analyse the information upon which this comparison takes place.

2. *Planning is interleaved with reasoning about preferences*. Preferences are often context-dependent and conflicting preferences contribute towards the growing complexity of collaborative planning and the decision-

making processes. In order to create balanced proposals, conflicting preferences between SC trading partners need to be considered as early as possible during the planning phase for further deliberation to find a consensus over a course of action. This process is known as meta-argumentation. During meta-argumentation, reasoning and supporting data are provided by each member to support their viewpoint, and they agree or disagree by providing a justification for their opinion. The re-configuration of positions as a result of meta-argumentation between members strengthens each other's cumulative contribution towards practical planning.

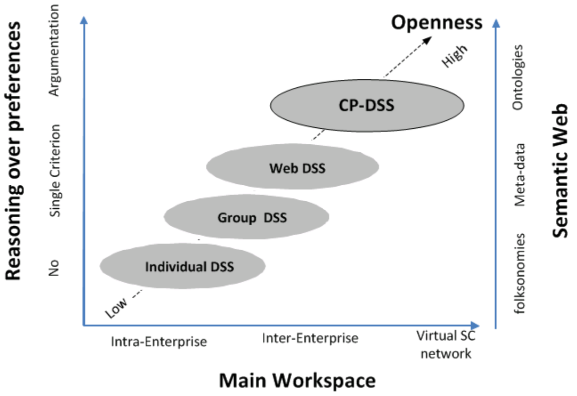

Figure 1: Evolution of Decision Support Systems

2 State-of-the-art.

The collaboration problem in decentralized SCs consists of two interleaving functions, namely planning and decision support. Planning involves generating plans whose success is warranted by some evidence coming from either one or different SC trading partners and decision support to evaluate the acceptability of these plans by comparing the evidence supporting them against possible objections. A planner needs to reason about their actions. Therefore, during collaborative planning, decision makers need decision support models for choosing, organising, and revisiting their actions and plans [7]. There is a plethora of work on planning and decision support in supply chains which can be divided into the following three categories:

2.1 Mathematical and analytical based approaches

Several supply chain collaboration practices, such as the Vendor Managed Inventory, Just In Time, Efficient Consumer Response, Continuous Replenishment and Accurate Response, Collaborative Planning, Forecasting and Replenishment (CPFR) that have been suggested in the literature focus on better planning through tight processes, integration and sharing the forecasting information among the SC trading partners [8]. However, in these applications, either no or very little decision-making aid is provided with respect to the negotiation process. Several researchers introduced a process model concerned with the decision-making and negotiation aspect of collaborative planning between trading partners while respecting their local decision-making authority [9; 10]. The research approaches discussed above are drawn from applied mathematics, such as optimization, statistics and decision theory and act as a black box for decision makers. They take in information and generate results but provide no visibility as to the underlying reasoning and justification behind the results. As a result, there is limited adoption of such approaches in SCs [11].

2.2 Logic-based approaches for automated planning and decision making

Multi-agent planning systems are advanced planning systems (APS) applied to independent or loosely-coupled problems to enhance the benefit of distributed planning between autonomous software agents [12]. During coordination, software agents who are engaged in collaboration build a model of other agents' mental states and update their own beliefs and goals as the dialogue progresses[13]. To handle uncertainty, argumentation-driven frameworks that allow different software agents to share their knowledge and resolve conflicts between them to reach the common goal have been proposed [12]. However, in most APS systems, planning provides the solution, and on execution it merely traverses the identified path. Such systems work well in the Closed-World Assumption where all the possible effects of each action are known in advance. However, planning becomes very challenging if the environment is dynamically changing (the Open-World Assumption) and is not pre-engineered to conform to software agent's needs. Furthermore, planners want to use tools for better visibility of the planning process but want to control the decision-making part of the planning phase.

2.3 Logic based approaches supporting planning and decision making

The introduction of Semantic Web technology tools for collaboration has addressed some of the issues of collaboration among SC trading partners, such as information has meaning attached to it that makes it understandable across organisational boundaries and facilitates data sharing and integration [14]. Attempts have been made to represent incomplete and contradictory information in information systems such as Dr-Prolog, Dr-Device, and Situated Courteous logic [15]. These implementations only represent and handle individual conflicting preferences by defining priorities based on a single criterion between them before engaging in collaboration. Therefore, these attempts do not provide a solution for collaborative planning that is subject to inconsistencies that derive from multiple data/information sources and multiple users. Furthermore, these techniques have not yet been applied to collaborative planning and decision support domains. The Collaborative Planning and Acting Model [16] is the first attempt to support planners in managing and planning information and facilitates the planning process with automated reasoning. However, the model lacks the means to represent incomplete and contradictory information and logical relations that define constraints and the axioms of the domain being modeled.

This research is first of its kind to study collaborative planning and decision support for practical reasoning in decentralized SCs. It aims to better align the activities of SC trading partners to overcome the operational risks as shown in Figure 1. The challenge is two-fold: *firstly*, how to take into account incomplete and contradictory planning information, reason and integrate it for proactive risk prediction and better planning; *secondly*, the current logic-based approaches supporting planning and decision making only handle individual preferences in the form of priorities. Additionally, the use of these priorities is usually embedded in the reasoning mechanism and competing rules are compared individually during the reasoning process. Therefore, the derivation notion is bound to one single comparison criterion. In such a scenario, the explanation of the results is based on a single criterion only and fails to take into account the multiple factors important for decision-making.

3 Motivation

Despite the increasing use of advanced planning tools and technologies, the current SC trading partners are far from structured collaborative planning and decision support for practical reasoning. Of the various factors, data integration has been considered as one of the core problems in business information systems [17]. In the last decade, the focus of SC trading partners was on *Systems of Records* (SoR) i.e. structured data about sales, customer and product information, inventory forecasts and so, and it was used for planning and decision making. As a result, centralized enterprise information systems such as data warehousing systems, exclusively dealt with record-oriented data that was carefully mapped using schema-centric mediation approaches by knowledge experts to support planning decisions. In such information systems, the business intelligence derived for planning can often provide decision support by answering the questions related to patterns of "*what is happening*". To answer questions related to patterns of "*why is it happening and what is the rationale behind it*", it is necessary to conjointly mine the SoR with the information generated as a result of the *Systems of Engagement* (SoE) with business partners or customers. The SoE are more decentralized, incorporate digital technologies for peer-to-peer interactions, and enable SC members in a network to collaborate and engage across a range of pivotal transactional processes on a global scale as shown in Figure 2. Therefore, SoE information complements SoR data with better insight, reason and interpretation. For example, by using SoR, a supplier can predict SC disruptions that may be caused by unexpected demand patterns from other trading partners through predictive analytics, however, they would not able to obtain information on the nature of complaints or requests made by the trading partner during the negotiation process which are not captured and stored as SoE in the repositories' holding emails or transcribed phone call information records during the planning and decision-making process. Therefore, this problem of knowledge sharing and practical reasoning involves creating SoE along with SoR and uses them conjointly to find patterns, knowledge and relationships for better decision support during collaborative planning to overcome the issue of information asymmetry. Practical reasoning is reasoning about what is to be done and it includes some actions or the adoption of an intension to initiate a sequence of actions later [1]. It underlines the need for informed and experienced judgment in many situations for decision support applications in SCs.

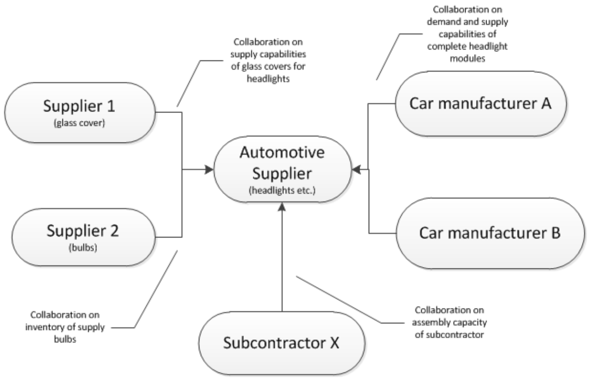

Figure 2: Collaboration in decentralized supply chain

4 Proposed conceptual framework

In this section, the solution for Collaborative planning and Decision Support in Decentralized Supply Chains is proposed to aligned the activities of trading partners in a supply chain. The proposed framework uses Defeasible logic programming (DeLP) as knowledge representation and reasoning language [18]. DeLP which is general-purpose defeasible argumentation formalism based on logic programming, is used to model inconsistent and potentially contradictory knowledge. A defeasible logic program has the form ψ= (Π, Δ), where Π and Δ stand for strict knowledge (non-defeasible) and defeasible knowledge (tentative), respectively. DeLP uses the argumentation formalism for reasoning over contradictory information by identifying conflicting information in the knowledge base and applying the dialectical process to decide which information prevails during the argumentative reasoning process. DeLP only uses goal-driven reasoning with the objective of serving only the users queries. It does not provide a solution for data-driven reasoning to infer new knowledge from existing information [19]. In my previous work [20], DeLP has been extended in order to make it suitable for information representation and reasoning in a Semantic Web application. The extensions made are as follows :

- Defined syntax and semantics for DeLP to represent business (planning) rules for hybrid reasoning.
- Proposed an argumentative production system that uses DeLP as the information representation language, and performs hybrid reasoning over incomplete and/or contradictory information.

In this research, I use extended DeLP for Semantic Web application to represent the planning tasks in the CP-DSS. Figure 2 depicts the proposed framework for CP-DSS and the next sub-sections outline the functionality of framework in detail.

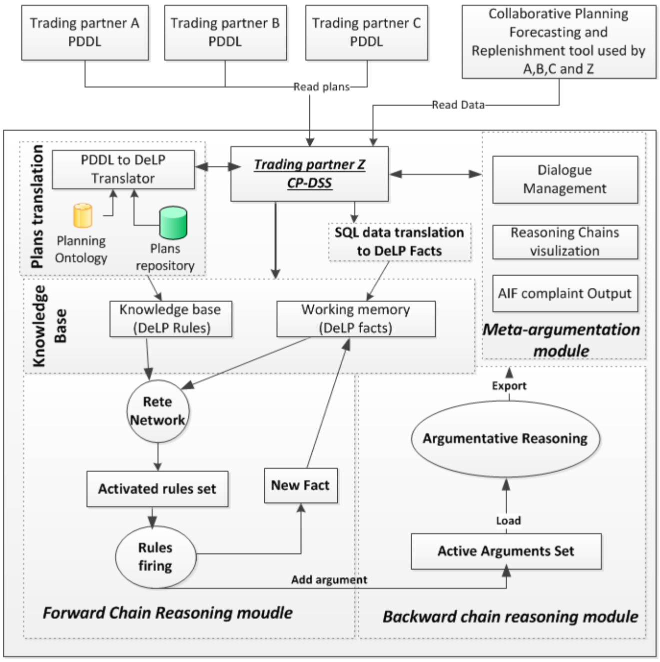

Figure 3: Conceptual framework for CP-DSS

4.1 Translating PDDL to DeLP

The PDDL (Planning Domain Description Langauge) is a standard language to describe real world planning domains [21]. Various tools such as GIPO [22] and itSIMPLE [23] provide graphical user interface to planning experts to model the real-world problems and export the plans in PDDL format so that they can be consumed by decision support tools and automated planners.

A PDDL planning problem is described in two sections: domain definition and problem specification. The domain describes all the elements which characterize the domain for planning, i.e., object types, predicates, actions (by specifying their inputs, outputs, preconditions and effects), etc. The problem essentially describes initial and goal states, by specifying the set of predicates assumed to be true in the initial state and the set of predicates to be satisfied in the goal state.

Figure 4 depicts the translation mapping from PDDL to DeLP constructs using PDDL-DeLP translator and algorithm 1 shows the entire process of the translation to DeLP constructs from PDDL in a higher level algorithm language. The first step is to download the trading partners PDDL files. During this process, the translator reads the PDDL files and saves certain information about the planning tasks such as file URL, owner/creator of planning tasks, download date etc., and saves the information in a database for their profiling. Once the download is complete, the next step is translation of PDDL files to DeLP rules and facts and save them in the knowledge base.

In the knowledge base, a planning task takes the following form: *[rule identifier][rule body] [type of rule] [head]*. The rule body represents the precondition and rule head represent the effects. There are two types of rules supported b the systems i.e., strict rule represented by solid arrow and defeasible rule represented by dotted arrow. During translation, a planning action (DeLP rule) that is in conflict with another planning action (DeLP rule) in the knowledge base is represented as defeasible rule while rest of the DeLP rules are represented as strict rules. Once the translation is complete, the CP-DSS also reads the relevant SQL data from CPFR and translate them into DeLP facts, stores them in the knowledge base.

PDDL	DeLP Rule	DeLP Predicate
(:types X -person)		Person(X)
(:predicates (shopper ?X – person))		Shopper(X) Person(X)
(:predicates (bulkOrder ?per – person ?item - Items))		bulkOrder(PER,ITEM) Person(PRE), Items(ITEM)
(:action ordinaryDiscount-supplier :parameters (?y - items ?x - supplier) :precondition (and (gstFree ?y) (giveDiscount ?x)) :effect (ordinaryDiscount ?x)	[s2] gstFree(Y), giveDiscount(X) → ordinaryDiscount(X)	
(:action notgstFree-prodct :parameters (?z - shop ?y- items) :precondition (and (eShop ?z) (packaging ?y)) :effect (not (gstFree ?y)))	eShop(Z), not packaging(Y,Z) ...> ~gstFree(Y)	

Figure 4: Translation mapping from PDDL to DeLP constructs

```
Data: PDDL files {a,b,c}
Result: DeLP Knowledge base
Array trans []= {a, b, c};
int i=0;
foreach trans.length do
    pddlfile[i].profiling(trans[i]);
    foreach action in pddlfile[i] do
        Array actions[] = readActions(action);
        foreach action in actions[i] do
            Array predicates=create(predicate);
            Array types=createtypes;
            Array constants=create(constants);
            Array delpRules=create(predicates,types,
            constants, action[i])
        end
        i++;
    end
end
i =0;
CreateKnowledgeBase(delpRules);
```

Algorithm 1: Translation of PDDL file to DeLP knowledge base

Considering supplier chain scenario discussed in figure 2, the trading partner z downloads the trading partners PDDL files, translated and resulted into ψ as follows:

$$\left\{\begin{array}{l}
[a.d1] shopper(X), product(Y), not\ advancePyament(X, Y) \\
\dashrightarrow \sim giveDiscount(X) \\
[b.d2] shopper(X), purhcase(X, Y), \\
bulkOrder(X, Y) \dashrightarrow giveDiscount(X) \\
[a.d3] eShop(Z), packaging(Y, Z) \dashrightarrow gstFree(Y) \\
[c.d5] eShop(Z), not\ packaging(Y, Z) \dashrightarrow \sim gstFree(Y) \\
[z.s2] gstFree(Y), giveDiscount(X) \\
\rightarrow ordinaryDiscount(X) \\
[z.s1] not\ gstFree(Y), giveDiscount(X) \\
\rightarrow normalDiscount(X) \\
[z.d7] shopper(X), normalDiscount(X) \\
\dashrightarrow platinumDiscount(X) \\
[c.d8] shopper(X), normalDiscount(X), \\
plansSlowToPay(X) \dashrightarrow \sim platinumDiscount(X)
\end{array}\right\}$$

[a.d1] is rule identifier where ‘a’ represents the trading partner and ‘d1’ represents the rule identifier.

4.2 Hybrid reasoning for arguments construction and conflicts handling

Once the translation from PDDL to DeLP is complete then the next step is arguments construction. This step is further divided into the following two sub-steps: Firstly, compilation of DeLP rules as a Rete network; Secondly, data driven reasoning for arguments construction.

In the proposed framework, the general Rete network [24] has been extended to represent incomplete and/or contradictory information as Rete nodes in the network. The extensions made to one-input nodes are as follows:

- AssertCondition: The one-input nodes have been extend to represent contradictory information by introduction strong negation i.e.$\sim$, as an attribute in the AssertCondition class.
- NegativeConditionNAF: A new type of one-input node was introduced to indicate incomplete information represented by the symbol ‘not’.

To explain the compilation of DeLP rules in a Rete network, consider the rule base ψ outlined in section 4.1 and its subset of rules compilation represented in fig. 6 in the form of a Rete network. The predicates that make up the body of the planning rules such as bulkOrder(X,Y), shopper(X) etc are represented as one input node and the claim of the DeLP rules such as a.d1, b.d2 and a.d3 are depicted as terminal nodes. The nodes in between the one-input node and the terminal nodes are represented as two-input nodes.

Once the compilation of DeLP as a rete network is complete, the next step is to perform data-driven reasoning over underlying information in the knowledge base by passing the DeLP facts in the working memory through the Rete network. Data-driven reasoning is a forward chain reasoning that starts by the introduction of DeLP facts in the Rete network. This results in the activation and firing of the DeLP rules. The derived DeLP facts flow back into the Rete network which, in turn, results in the activation of new DeLP rules. This process continues until no more rule/s are activated. During this process, the activated DeLP rules are saved arguments in an arguments set. It is important to note here is that if the activated DeLP rules’ body represents some predicate starting with the symbol ‘not’, then before its firing, a query is sent to the DeLP server to compute its truthfulness by querying the knowledge base. If the query returns yes, then the DeLP rule is fired, otherwise the activated DeLP rule will be removed from the activated rule set. It is important to note here is that in current research rete network has been extended from the single rule execution strategy to execute all rules that are activated during data-driven reasoning. Taking into consideration the scenario depicted in figure 3, forward chain reasoning is used to digitize the planning tasks and make them alive for the planners as shown in the following illustration:

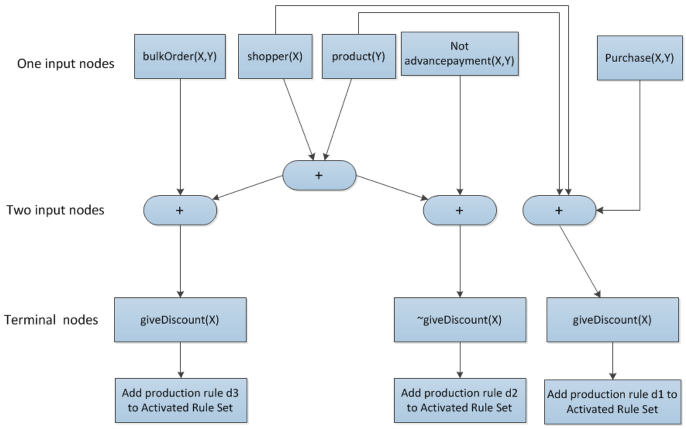

Figure 5: Compilation of production rules in the form of a Rete network

$$\left\{\begin{array}{l}
[a.d1] shopper(david), purchase(david, rawMaterial) \\
\dashrightarrow giveDiscount(david) \\
[b.d2] shopper(david), notadvancePayment(david, rawMaterial) \\
\dashrightarrow \sim giveDiscount(david) \\
[a.d3] shopper(david), purchase(david, rawMaterial), \\
bulkOrder(david, rawMaterial) \dashrightarrow giveDiscount(david). \\
[c.d5] eShop(BigW), not\ packaging(BigW, rawMaterial) \dashrightarrow \\
\sim gstFree(rawMaterial) \\
[z.s1] notgstFree(rawMaterial), giveDiscount(david) \\
\rightarrow normalDiscount(david) \\
[z.d7] shopper(david), normalDiscount(david) \\
\dashrightarrow platinumDiscount(david) \\
[c.d8] shopper(david), product(rawMaterial), \\
havefeedback(rawMaterial, feedback), \\
reviewRate(feedback, good) \dashrightarrow purchase(david), rawMaterial)
\end{array}\right\}$$

It is important to note here is that the DeLP rules are initiated with the domain knowledge defined in the descriptive model and the DeLP facts imported from CPFR. Once the arguments construction is complete, the next step is conflicts identification and their resolution using goal-driven reasoning. Conflicts identification and their resolution is a recursive process consisting of the following two steps:

- Identification of an argument and its counter-argument.
- Compute priority between conflicting arguments by building and marking of dialectical trees as depicts in Algorithm 2.

(Dialectical tree [18]) If an argument $\mathcal{A}$ counter-argues argument $\mathcal{B}$, and no static defeat exists, then we construct a dialectical tree for argument $\mathcal{A}$ to determine whether argument $\mathcal{A}$ defeats argument $\mathcal{B}$ or vice versa.

Let $\mathcal{A}$ be an argument. A dialectical tree for argument $\mathcal{A}$, is $\Sigma(\mathcal{A}$, h) where h is claim($\mathcal{A}$), is recursively defined as follows:

(1) A single node labeled with an argument ($\mathcal{A}$, h) with no counter-argument is by itself a dialectical tree for ($\mathcal{A}$, h). This node is also the root of the tree.

(2) Suppose that $\Sigma(\mathcal{A}$, h) is an argument with counter-arguments ($\mathcal{A}_1$,h_1), ($\mathcal{A}_2$,h_2),.........,($\mathcal{A}_n$,h_n), we construct the

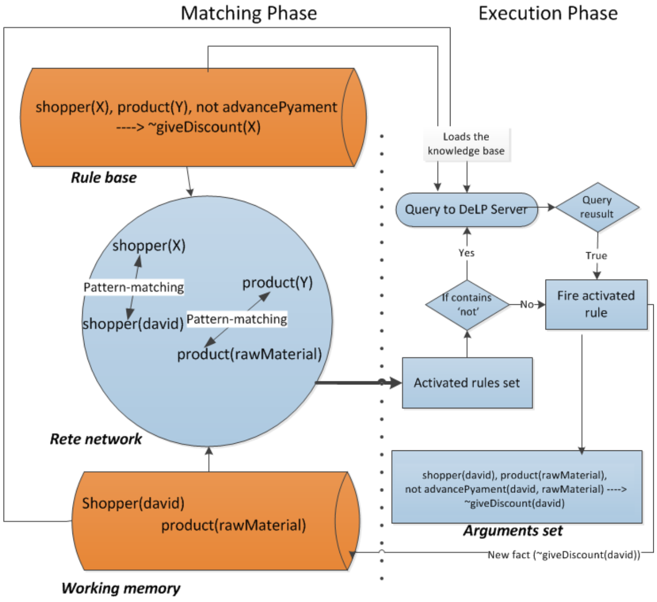

Figure 6: Data-driven reasoning by passing the facts through the Rete network

dialectical tree for $(\mathcal{A}, \text{h})$, $\Sigma(\mathcal{A}, \text{h})$ by labeling the root node with $(\mathcal{A},\text{h})$ and by making this node the parent of the root of dialectical trees for $(\mathcal{A}_1,\text{h}_1)$, $(\mathcal{A}_2,\text{h}_2)$,.........,$(\mathcal{A}_n,\text{h}_n)$ i.e. $\Sigma(\mathcal{A}_1, \text{h}_1)$,$\Sigma(\mathcal{A}_2, \text{h}_2)$,........,$\Sigma(\mathcal{A}_n, \text{h}_n)$.

(Marking of dialectical tree [18]):

- Leaves of $\Sigma(\mathcal{A}, \text{h})$ are U-nodes.
- Let $(\mathcal{B}, \text{q})$ be an inner node of $\Sigma(\mathcal{A}, \text{h})$. Then (B, q) will be a U-node iff every child of $(\mathcal{B}, \text{q})$ is a D-node. The node $(\mathcal{B}, \text{q})$ will be a D-node if it has at least one U-node as a child.

If dialectical tree is marked as undefeated then this will trigger the process of meta-argumentation which is described in section 4.3. Once the hybrid reasoning is complete, the CP-DSS will display the reasoning results in a graphical format so that decision makers are able to better comprehend the results.

(Reasoning Chain): An argument $\mathcal{A}$ supported by a chain of sub-arguments produces a reasoning chain $\lambda_{\mathcal{A}}$= $(\mathcal{A}_1, ...,\mathcal{A}_n)$ for an argument $\mathcal{A}$. The claim of supported argument $\mathcal{A}$, is called a 'result' of the reasoning chain and the chain of sub-arguments is called a 'support' for the result of the reasoning chain and is define as follows: $\forall r,\ s \in Args$ $\{\ if(\ s\ \xi\ r)\ then\ \lambda_{(r,j)}$= $\lambda_{(r,,j)} \cup\ s$ where ξ is used to represent sub-argument relationship and $\lambda_{(r,j)}$ is used to represent a reasoning chain with result j. Algorithm 3 outline the process of reasoning chains construction. To share the reasoning chain with other Semantic Web applications, the reasoning chains will be annotated with an argument interchange format (AIF)[25] and is shared with trading partners in RDF\XML format.

```
Data: $(\mathcal{A}, h)$
Result: $\Sigma_{status}(\mathcal{A}, h)$
Let C ← get all counter-arguments of $(\mathcal{A}, h)$;
if $C \neq \emptyset$ then
    while there is no $\Sigma_U(\mathcal{A}_i, h_i) \in C$ do
        for every argument in C do
            Let $(\mathcal{A}_i, h_i)$← minimal non-labelled element
            BuildDialecticalTree($(\mathcal{A}_i, h_i)$) getting result as $\Sigma(\mathcal{A}_i, h_i)$;
            Put $\Sigma(\mathcal{A}_i, h_i)\ \xi\ (\mathcal{A}, h)$
        end
        if there exist some $\Sigma_U(\mathcal{A}_i, h_i)$ then
            Set $\Sigma_D(\mathcal{A}, h)$;
        else
            Set $\Sigma_U(\mathcal{A}, h)$;
        end
    end
else
    $\Sigma(\mathcal{A}, h)$= $(\mathcal{A}, h)$;
    Set $\Sigma(\mathcal{A}, h) \leftarrow defeated$;
end
```

Algorithm 2: Building and marking of Dialectical trees

4.3 Interleaving planning and reasoning over conflicting preference using meta-argumentation

Meta-argumentation is a deliberation dialogue that involves participants who share responsibility and collaborate on deciding what action or course of actions should be undertaken in a given situation [26]. In such dialogues, participants don't have fixed positions at the start of the dialogue and the goal and need for action can originate from any of the various participants involved. During the course of action, however, participants may be involved in a persuasion dialogue which may motivate them to model a persuasion dialogue as embedded in a deliberation dialogue. In particular, the following tasks will be performed in this sub-aim:

A plethora of work exists on building dialogue-based systems for software agents. This research focus on extending the work done by [27] using argumentation schemes [28]. During the process of argumentation, relationships between the arguments are linked with each other in a certain pattern to support the ultimate conclusion. Such linking patterns are called 'argumentation schemes' and allow reasoning to be performed using a set of premises and a conclusion. These argumentation schemes have emerged from informal logic [29]. The schemes help to categorize the way that arguments are built. They bridge the gap between logic-based application and human reasoning by capturing stereotypical patterns of human reasoning. An example is an argument from an expert opinion scheme. Formally, an argumentation scheme is composed of a set of premises A_i, a conclusion denoted as S, and a set of critical questions CQ_i is aimed at defeating the derivation of the consequent. In this research, arguments are built using argumentation schemes during meta-argumentation. The objective is two-fold, firstly; to enable planners to put forward their arguments that may be incomplete statements and offer them ways of advancing well-formed arguments as well as to reuse arguments that often appear in discussions; secondly, with the help of algorithms, to compute the acceptability of arguments at any stage of the discussion.

```
Data: ($\mathcal{A}$, $h$)
Result: $\lambda_{(\mathcal{A},h)}$
Let S ⟵ get all sub-arguments of ($\mathcal{A}$, $h$);
if $S \neq \emptyset$ then
    foreach ($\mathcal{A}_i$, $h_i$) $\in$ S do
        if noCounterArgument($\mathcal{A}_i$, $h_i$) or $\Sigma_U$($\mathcal{A}_i$, $h_i$) then
            BuildReasoningChain(($\mathcal{A}_i$, $h_i$)) ;
            Put $\lambda$($\mathcal{A}_i$, $h_i$) $\xi$ ($\mathcal{A}$, $h$);
        end
    end
else
    $\lambda_{(\mathcal{A},h)}$ = ($\mathcal{A}$, $h$);
end
```

Algorithm 3: Construction of a reasoning chain

The deliberation dialogue system is defined by:

1. Topic Language: DeLP as a logical language.
2. Argumentation Logic: as defined in [15]. The only difference is that in our previous work it was assumed that the system has collated all the relevant information and reasoning engine reasoning over it. In this system, human planners are collaborating and conflict resolution process is a dialogue -driven activity. We reuse the definition of argument, sub-argument, attack, static defeat and dynamic defeat.
3. Communication Language to define set of Locutions S and two binary relation Ra and Rs of attacking and surrendering reply on S. Dialogue moves and termination as defined in [27].

To answer the questions of a decision maker which may help him to understand the reasoning process (that is, to obtain an explanation on the conclusion achieved), CP-DSS provides a querying mechanism to query the knowledge base. A query 'q' consists of a predicate, and can be executed on the argument set 'Args' with the help of function executeQuery(q) to check the support for the predicate in the argument set and returns the dialectical tree. Taking into considering a supply chain scenario depicts in figure 3, the figure 7 depicts the two conflicting reasoning chains produced by hybrid reasoning engine that will call for meta-argumentation in order to proceed for an action.

4.4 Prototype development and future work

The development of to address the requirements of different collaborative planning for pratical reasoning in supply chains is carried out with help Microsoft Visual Studio 2010[1], NRuler [2] whis is a fast production system library based on the RETE algorithm, written in C sharp. This library is extended for the development of the hybrid reasoning engine. QuickGraph [3] that provides generic directed/undirected graph data structures and algorithms for .NET. It also supports Graphviz [4] to render the graphs. It is used to generate the graphical representation of the reasoning results produced by theCP-DSS and DeLP Server that is is an implementation of defeasible logic programming (DeLP). It is used as a back-end server for the development of the hybrid reasoning engine. MySQL [5] open source relational database for storing PDDL profiling information.

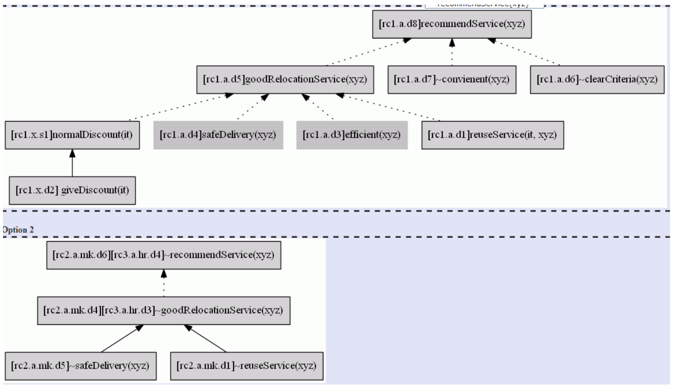

Figure 7: CP-DSS depicting graphical representation of conflicting plans to start meta-argumentation

In future, CP-DSS will be extend with Information sharing and integration for proactive operational risk prediction. In a static environment, SC members may choose to share specific and efficient process linkages and information sharing/exchange mechanisms with selected partners. However, in a dynamic environment SC business partners need to develop more robust and reconfigurable digital linkages that can deal with changes in the business environment [30]. For example, a SC works as a network and when some event such as operational risk occurs, it is not the entire supply chain that needs to deal with this event. Once the semantic-annotated information is shared, it is used for proactive risk identification to reduce uncertainty. Uncertainty in this context is a lack of knowledge regarding the occurrence of an event in a SC to overcome operational risk. Using information for event detection requires the identification of the correlation between events and shared information among SC members. The idea is how to relate concepts to a certain event and predict the occurrence of the event using shared information.

References

[1] R. Audi, "A theory of practical reasoning," *American Philosophical Quarterly*, vol. 19, no. 1, pp. 25–39, 1982.

[2] G. Moore, "Systems of engagement and the future of enterprise IT-a sea change in enterprise IT," *Association for Information and Image Managment, Silver Spring*, 2011.

[3] A. L. Council, "The economic significance of the Australian Logistics Industry," tech. rep., Australian

---

[1] http://www.microsoft.com/visualstudio/en-us

[2] http://nruler.codeplex.com/

[3] http://quikgraph.codeplex.com/

[4] http://www.graphviz.org/

[5] http://www.mysql.com/

Logistics Council, http://austlogistics.com.au/wp-content/uploads/2014/07/Economic-Significance-of-the-Australian-Logistics-Indsutry-FINAL.pdf, 2014.

[4] R. Spekman and E. W. Davis, "The extended enterprise: a decade later," *International Journal of Physical Distribution & Logistics Management*, vol. 46, no. 1, pp. 43–61, 2016.

[5] T. S. Dillon, Y.-P. P. Chen, E. Chang, and M. Mohania, ch. Conjoint Mining of Data and Content with Applications in Business, Bio-medicine, Transport Logistics and Electrical Power Systems, pp. 1–19. IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg, 2014.

[6] Y. Daultani, S. Kumar, O. S. Vaidya, and M. K. Tiwari, "A supply chain network equilibrium model for operational and opportunism risk mitigation," *International Journal of Production Research*, vol. 53, no. 18, pp. 5685–5715, 2015.

[7] M. Ghallab, D. Nau, and P. Traverso, "The actor's view of automated planning and acting: A position paper," *AI*, vol. 208, no. 0, pp. 1–17, 2014.

[8] Y. Sheffi, "The value of CPFR," in *Proceedings of the Fourth International Congress on Logistics Research, IMRL, Lisbon, Portugal*, 2002.

[9] Y. Dong, X. Huang, K. K. Sinha, and K. Xu, "Collaborative demand forecasting: Toward the design of an exception-based forecasting mechanism," *Journal of Management Information Systems*, vol. 31, no. 2, pp. 245–284, 2014.

[10] G. Dudek, *Collaborative planning in supply chains: a negotiation-based approach*, vol. 533. Springer Science & Business Media, 2013.

[11] E. J. Umble, R. R. Haft, and M. M. Umble, "Enterprise resource planning: Implementation procedures and critical success factors," *European journal of Operational Research*, vol. 146, no. 2, pp. 241–257, 2003.

[12] A. Torreño, E. Onaindia, and Ó. Sapena, "An approach to multi-agent planning with incomplete information.," in *ECAI*, pp. 762–767, 2012.

[13] A. Gabaldon and P. Langley, "Dialogue understanding in a logic of action and belief," *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[14] N. K. Janjua, *A defeasible logic programming-based framework to support argumentation in Semantic Web applications*. Springer Science & Business Media, 2014.

[15] N. K. Janjua and F. K. Hussain, "Web@ IDSS-argumentation-enabled web-based IDSS for reasoning over incomplete and conflicting information," *Knowledge-Based Systems*, vol. 32, pp. 9–27, 2012.

[16] J. Patel and M. Dorneich, "Improving coalition planning by making plans alive," *Intelligent Systems, IEEE*, vol. 28, pp. 17–25, Jan 2013.

[17] G. C. Stevens and M. Johnson, "Integrating the supply chain 25 years on," *International Journal of Physical Distribution & Logistics Management*, vol. 46, no. 1, pp. 19–42, 2016.

[18] A. J. Garcia and G. R. Simari, "Defeasible logic programming: an argumentative approach," *Theory and Practice of Logic Programming*, vol. 4, no. 1+2, pp. 95–138, 2004.

[19] N. K. Janjua, O. K. Hussain, F. K. Hussain, and E. Chang, "Philosophical and logic-based argumentation-driven reasoning approaches and their realization on the WWW: A survey," *The Computer Journal*, vol. 58, no. 9, pp. 1967–1999, 2015.

[20] N. K. Janjua and F. K. Hussain, "Web@idss @ argumentation-enabled web-based idss for reasoning over incomplete and conflicting information," *Knowledge-Based Systems*, no. 0, pp. –, 2011.

[21] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "Pddl-the planning domain definition language," 1998.

[22] R. M. Simpson, D. E. Kitchin, and T. McCluskey, "Planning domain definition using gipo," *The Knowledge Engineering Review*, vol. 22, no. 02, pp. 117–134, 2007.

[23] T. S. Vaquero, J. R. Silva, M. Ferreira, F. Tonidandel, and J. C. Beck, "From requirements and analysis to pddl in itsimple3. 0," *Proceedings of the Third International Competition on Knowledge Engineering for Planning and Scheduling, ICAPS 2009*, pp. 54–61, 2009.

[24] C. L. Forgy, "Rete: A fast algorithm for the many pattern/many object pattern match problem," *Artificial Intelligence*, vol. 19, no. 1, pp. 17–37, 1982.

[25] I. Rahwan and C. Reed, "The argument interchange format," in *Argumentation in Artificial Intelligence* (I. Rahwan and G. R. Simari, eds.), ch. 19, pp. 383–402, Springer, 2009.

[26] D. N. Walton and E. C. Krabbe, "Commitment in dialogue: Basic concepts of interpersonal reasoning.," *State University of New York Press*, vol. 35, 1995.

[27] E. M. Kok, J.-J. C. Meyer, H. Prakken, and G. A. Vreeswijk, "A formal argumentation framework for deliberation dialogues," in *Argumentation in Multi-Agent Systems*, pp. 31–48, Springer, 2011.

[28] N. K. Janjua, F. K. Hussain, and O. K. Hussain, "Semantic information and knowledge integration through argumentative reasoning to support intelligent decision making," *Information Systems Frontiers*, vol. 15, no. 2, pp. 167–192, 2013.

[29] D. J. Godden and D. Walton, "Advances in the theory of argumentation schemes and critical questions," *Informal Logic*, vol. 27, pp. 267–292, 2007.

[30] S. Fayezi, A. Zutshi, and A. O'Loughlin, "Understanding and development of supply chain agility and flexibility: A structured literature review," *International Journal of Management Reviews*, pp. 1–30, 2016.