PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 23/08

A Four-sided View of Plot Composition

Antonio L. Furtado

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 23/08 ISSN 0103-9741
Editor: Prof. Carlos José Pereira de Lucena April 2008

A Four-sided View of Plot Composition*

Antonio L. Furtado

furtado@inf.puc-rio.br

Abstract: This paper argues that the process of plot composition can be viewed under a four-sided perspective, induced by the presence of syntagmatic, paradigmatic, antithetic and meronymic relations between the constituent events. In turn, these relations are shown to be associated with the four major tropes of semiotic research. A set of facilities for interactive plot composition and adaptation dealing with these four relations is described. To accomodate antithetic relations, corresponding to the irony trope, our plan-based approach leaves room for the unplanned. To illustrate the discussion, as well as the description of a logic programming prototype implementation, we use a small number of events, which, in strikingly different combinations, have been treated repeatedly in literary works.

Keywords: storytelling, plots, plan-generation, narratology, tropes.

Resumo: Este artigo argumenta que o processo de composição de enredos pode ser visualizado sob uma perspectiva quadri-lateral, induzida pela presença de relações sintagmáticas, paradigmáticas, antitéticas e meronímicas entre os eventos constitutivos. Tais relações, por sua vez, são mostradas em associação com os quatro tropos principais da pesquisa semiótica. É descrito um conjunto de facilidades para a composição e adaptação interativa de enredos, lidando com essas quatro relações. Para acomodar relações antitéticas, que correspondem ao tropo da ironia, nossa abordagem baseada em planos deixa espaço para o não planejado. Para ilustrar a discussão, bem como a descrição de um protótipo implementado em programação em lógica, usamos um pequeno número de eventos os quais, em combinações marcadamente diferentes, têm sido tratados em obras literárias.

Palavras-chave: narração de estórias, enredos, geração de planos, narratologia, tropos.

* This work has been partly sponsored by the Ministério de Ciências e Tecnologia da Presidência da República Federativa do Brasil.

In charge of publications

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br

Souvent femme varie,
Bien fol est qui s'y fie!
Victor Hugo, *Le Roi s'amuse*

1. Introduction

Narratology studies [Ba] distinguish three levels in literary composition: *fabula*, *story* and *text*. At the fabula level, the *characters* (also known as *dramatis personae*) acting in the narrative are introduced, as well as the narrative *plot*, consisting of a partially-ordered set of *events*.

In the present, work, we stay at the fabula level and give special attention to plots whose constituent events are normally made to happen as a consequence of a predefined repertoire of actions, which we shall call *operations*, deliberately performed by the characters. Since we are interested in the interactive composition of plots, we shall also consider the possibility of user interventions through certain permitted *directives*.

Throughout the presentation, we shall treat plot composition as a *plan generation* process, and hence the terms plot and plan shall be used interchangeably. A plan generator should be able to align the plot events in a *coherent sequence* in view of the characters' objectives, whenever possible coming up with more than one plot, so as to provide *alternative ways* to reach the objectives. But narratives are often more attractive when *unplanned shifts* are allowed to occur. This is arranged for in our proposal through the limited power given to users to interfere with the planner, causing certain discontinuities in the context, particularly concerning changes in the feelings and beliefs of certain characters. Finally, one should have the possibility to obtain from the planner a more *detailed account* of the events, by having them expanded into smaller grain actions.

It turns out that the need to consider these four notions – coherence, alternatives, transgressive shifts, details – informally prescribed in the above paragraph as desirable for any effective plot composition process, brings to mind four different types of relations between events: syntagmatic, paradigmatic, antithetic and meronymic, which in turn are associated with the so called *four major tropes* of semiotic research [Bu], namely metonymy, metaphor, irony and synecdoque.

Starting from such considerations, the present paper proposes a four-sided way to characterize plot composition at the fabula level. Section 2 describes the four relations between events, and points out their correspondence to the four major tropes. Section 3 is a brief literary survey intended to motivate the ubiquitous example that we chose for illustrating the discussion. Section 4 sketches the main features of our plan-based prototype tool. Brief conclusions are presented at section 5. Appendix A documents the example.

2. The four viewpoints

Early work in linguistics [Sa] has characterized two orthogonal dimensions in the structure of language, deployed along the so-called *syntagmatic* and *paradigmatic* axes. The notion is readily applicable to the events of which a narrative plot is composed, as we shall illustrate through an example to be referenced throughout the paper.

Consider four types of events, all of them having one woman and two men as protagonists: *abduction*, *elopement*, *rescue*, and *capture*. As Propp's seminal work [Pr] has

demonstrated, many plots mainly consist of a villainy, i.e. of a violent action that breaks the initially stable and peaceful state of affairs, followed ultimately by an action of retaliation, which may or may not lead to a happy outcome.

Here we shall restrict events to those resulting from the action of the characters, the *dramatis personae* typical of the chosen genre. Curiously the distribution of roles when the four events above are considered is not unique: we called the violent initial act a "villainy", but the author of abduction, and more often of elopement, can be the hero of the story, not the villain, and in such cases the woman's original guardian (husband, father) is regarded as the villain.

2.1. Syntagmatic relations

To declare that it is legitimate to continue a plot containing abduction by placing rescue next to it, we say that these two events are connected by a *syntagmatic relation*. More precisely, we can define the semantics of the two events in a way that indicates that the ocurrence of the first leaves the world in a state wherein the occurrence of the second is coherent. Similarly, a plot involving elopement followed by rescue looks natural, and therefore we may add that these two events are likewise related.

The syntagmatic relation between events induces a weak form of causality or enablement, that justifies their *sequential ordering* inside the plot.

2.2. Paradigmatic relations

Comparing the two plots above, we notice that the events of abduction and elopement are *alternative* ways to accomplish a similar kind of of villainy. Both achieve approximately – though not quite – the same effect: one man takes away a woman from where she is and starts to live in her company at some other place. There are differences, of course, since the woman's behaviour is usually said to be coerced in the first case, but quite voluntary in the second. In fact, it is usual to assume that a sentence such as "Helen elopes with Paris", implies that Helen very likely had fallen in love with Paris.

To express that abduction and elopement play a similar function, we say that there is a *paradigmatic relation* between the two events. Likewise, this type of relation is perceived to hold between the events of rescue and capture, which are alternative forms of retaliation. And, again, there is a difference between the woman's assumed attitude, associated as before with her feelings. An abducted woman expects to be rescued from the villain's captivity by the man she loves. On the contrary, if she freely eloped with the seducer, she will only leave him through forceful capture.

As the present example suggests, the orthogonality between the syntagmatic and the paradigmatic axes does not permit that the two relations be considered independently when composing a plot. Thus, in principle, the two pairs enumerated in the previous section (abduction-rescue and elopement-capture) are the *only* normal combinations, the only well-behaved sequences that can be generated through a discipline whereby the state of the world can be changed exclusively through certain predefined events, whose initiative is attributed to the participating characters. The next section, however, shows that this limitation can, and even should, be waived occasionally.

2.3. Antithetic relations

While normal plots, whose outcome is fully determined, can be composed exclusively on the basis on the two preceding relations, the possibility to introduce unexpected turns is sometimes desirable in order to make the plots more attractive – and this requires a third construct, which we chose to call *antithetic relation*. A context where a woman suffers abduction by a ravisher whom she does not love would seem incompatible with a capture event, since there should be no need to employ force to bring back the victim. So, in this sense, abduction and capture are in antithetic relation.

The occurrence of elopement followed by rescue provides an even stronger case of antithetic relation. Indeed, elopement only makes sense if the victim loves the seducer, whereas, for this very motive, she would resist to any attempt to rescue her, leaving forceful capture as the only viable alternative.

Generally speaking, if some *binary opposition* – the "to love or not to love" dilemma, in the present case – is allowed to be manipulated via some agency external to the predefined events, then one can have plots that no longer look conventional. A sort of discontinuity is produced by such radical shifts in the context. Intervening between abduction and capture, or between elopement and rescue, a sudden change of the woman's feelings can give rise to these surprising sequences.

A classic study of emotions, stressing interpersonal relationships [RO], attempted to formally characterize what is meant by a number of words and phrases expressing emotions, which are decomposed and analysed using a situation calculus formalism. Love and hate (or like and dislike) are not broken into simpler terms, being considered primitive and hence not explainable... What makes a woman change her feelings? What can a man do (or avoid to do) that may be expected to determine her reactions in the desired way? Well, some people, such as king François I^{er}, used to regard them as agents of uncertainty (cf. epigraph of section 1), but even a primitive aboriginal population in Australia recognized that there exist other members in the same category (which they call *balan*) of dangerous objects [La].

The blessings and hazards of love will be the main concern here, but other critical factors may dominate a narrative, depending on the chosen genre. Besides Boolean yes-or-no properties, binary oppositions can refer to any value range, e.g. from a low to a high measure for the strength of a given personage. Also, both in fiction and in reality, things not always proceed according to planned events. Natural phenomena and disasters, the mere passage of time, the intervention of agents empowered to change the rules, supernatural or magic manifestations, etc., cannot be discounted.

Specifically for the tragedy genre, Aristotle [Ar] distinguished simple from complex plots, characterizing the latter by the occurrence of *recognition* (αναγνορισισ) and *reversal* (περιπετεια). Recognition does not imply that the world itself has changed, but rather the *beliefs* of one or more characters about the actual facts. Because of a change of beliefs, a motive to be added to those enumerated in the previous paragraph, a reversal in the course of actions can take place, usually in a direction totally opposite to what was going on so far. Yet another possible external cause of both recognition and reversal in the tragic scene was the intervention of a god, who was lowered onto the stage using a crane – known, accordingly, as *deus ex machina*.

Aristotle's remarks are clearly relevant to the present discussion of plots in general. Following his lead, we shall admit state changes outside the regular regime of predefined events by allowing the user – literally acting *ex machina* (via the computer...) – to impose variations to the context, and thereby deviate the action from its predicted path.

This extreme device will be necessary to allow the elopement-rescue sequence. We decided, however, not to make it indispensable for abduction-capture, in order to to have a chance to present a good example of the effect of beliefs, which may wrongly contradict the actual facts. Criminal records everywhere are full of simulated abduction pacts, with the purpose of drawing a ransom from a deluded family. Conversely, a man can unnecessarily decide that capture is the only way to bring back a woman, if he mistankenly believes her in love with the ravisher.

2.4. Meronymic relations

Meronymy is a word of Greek origin, used in linguistics to refer to the decomposition of a whole into its constituent parts. Forming an adjective from this noun, we shall call *meronymic relations* those that hold between an event and a lower-level set of events, with whose help it is possible to provide a more detailed account of the action on hand.

Thus, we could describe the abduction of a woman called Sita by a man called Ravana as: "Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka." And her rescue by Rama could take the form: "Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace." But notice that such decompositions are not fixed, being comparable instead to context-sensitive compilation, since the lower-level operations are selected as required by the current state. For instance, with respect to the rescue event, the hero may already be present at the ravisher's dwelling, or perhaps the victim is not held in captivity, respectively obviating the need for the voyage or for fighting the enemy.

Detailing is most useful to pass from a somewhat abstract view of the plot to one, at a more concrete physical level, that is amenable (possibly after further decomposition) to the production of a computer graphics animation [CPFF].

The fact that the four high-level events correspond to very similar sequences of lower-level moves is revealing. It shows how easy it is to disguise a voluntary event as a forced one. Is the woman being abducted or is she, as previously concerted, eloping with the stranger? Simulated rapt is not a novelty in criminal records. Thus, moving up from a detailed to a summary description is no less useful, serving to help discover what is really happening.

Mixed plots, cobining events of different levels, do also make sense, satisfying the option to represent some events more compactly while showing the others in detail.

2.5. The four major tropes

Although it would be presumptuous to submit any finite set of concepts as sufficient to model a complex notion, such as that of narrative plots, there is at least one argument to suggest that the four relations studied here do encompass a relatively significant part of the problem. It turns out that the four relations correspond to what Kenneth Burke [Bu]

considers the four major tropes: *metonym*, *metaphor*, *irony*, *synecdoche*. In turn, it has been suggested that those rhetoric figures of speech provide models for remarkably comprehensive analyses in different areas [Cha,Wh].

The tropes are not defined in the same way by linguists, there being much disagreement, especially on the distinction between metonym and synecdoche. Here, we shall use a relatively well-accepted interpretation. An excellent discussion can be found in [Cha], where many practical applications of Burke's four tropes theory are surveyed.

Metaphor [LJ, Or] and synecdoque [Cha] have to do with hierarchical structures such as those represented in ontologies [BCT]. If one concept C_1 can be metaphorically used to denote another concept C_2, the two concepts are said to be similar or analogous, thus admiting a more general concept $\hat{C}$ subsuming both of them. C_1 and C_2 would be represented in the network with **is-a** links connecting them to $\hat{C}$. Also one could add an **is-like** link from C_1 to C_2 [BBFC]. Clearly, metaphor is a displacement along one of Saussure's axes, being thus suggestive of the *paradigmatic relation* between events.

In synecdoque, concept C_1 is used to denote concept C_2 if C_1 is a part of C_2 (which calls for another link, C_1 **part-of** C_2); the converse substitution, from whole to part, is also contemplated. The corresponding association between events is obviously what was called *meronymic relation* in the present paper.

According to [Cha], metonyms are based on various indexical relationships between concepts, notably the substitution of effect for cause. It conveys an idea of contiguity, in agreement with the s*yntagmatic relation* reviewed here, which justifies placing events in sequence.

Irony is the most intriguing of the four tropes. In [Cha], the notion is explored as follows: "Where it means the *opposite* of what it says (as it usually does) it is based on binary opposition. Irony may thus reflect the opposite of the thoughts or feelings of the speaker or writer (as when you say 'I love it' when you hate it) or the opposite of the truth about external reality (as in 'There's a crowd here' when it's deserted). It can also be seen as being based on substitution by *dissimilarity* or *disjunction*. Whilst typically an ironic statement signifies the opposite of its literal signification, such variations as understatement and overstatement can also be regarded as ironic. At some point, exaggeration may slide into irony." Disclosing paradoxes and hidden agendas in literary texts, sharp contrasts between the declared intentions and the real ones, is another source of irony, constituting a trend in critical studies known as *deconstruction* [Cu].

Not only mental attitudes, feelings and statements can be ironic – actions can also be ironic, but always in an unplanned, non-deliberate fashion [Bo]: "Irony is not limited to verbal acts, but is also a characteristic of situations that are often referred to as *dramatic irony* . . . I cannot say that I will do three ironic acts today because when I say that some act is ironic, I am asserting that it is somehow unexpected or inconsistent from my point of view, and I cannot claim this with respect to my own intentions."

Thus irony induces an *antithetic relation* between events that are, in principle, incompatible with each other, given their dependence on contexts characterized by radically opposite properties. Mediating two such events, the until then well-behaved world must suffer a disruptive shift, whereby the truth value of certain facts or beliefs is inverted, or certain properties move from one extreme to the other within the ascribed value range (e.g. from helplessly weak to heroically strong).

3. An example - dispute for a princess

As anticipated in the previous section, we shall illustrate the proposed four-sided view with stories wherein a woman is taken away from her guardian (husband, father or someone else of her family) by another man, and then recovered through the action of the former.

Let us first consider the "normal" plots, involving either:

a. abduction and rescue
b. elopement and capture

An example of (a) is the *Ramayana* of Vamiki [Va], a classic of the Sanskrit literature. We have already been using the names of its protagonists, and in fact have adopted these names for our running example to be presented in section 4. Princess Sita is abducted when alone in her temporary forest dwelling by the devilish Ravana, who imprisons her in his palace at the island of Lanka. Her heroic husband Rama makes war against the villain, finally succeeding to overcome him and rescue the princess.

The Irish *Story of Deirdre* [Mg] is a typical example of (b). The young Deirdre is forced to marry the elderly king Conchobar, but falls in love with Naoise and elopes with the young man. Conchobar tracks them down, Naoise is killed and Deirdre, shortly after being captured, commits suicide. In this story, as in several others, the roles are reversed, the eloper is clearly the hero whereas the husband plays the villain's part.

We now turn to the more problematic stories, involving one of the following sequences of events:

c. abduction and capture
d. elopement and rescue.

The mythical *Rape of the Sabines* shows what can happen as a consequence of a drastic reversal of the circumstances. King Romulus is facing a problem at the newly founded city of Rome: the population is entirely male at first. To remedy the lack, he leads his men to break into the houses of the Sabines and abduct their women. Sometime afterwards the Sabine warriors march against the Romans, but the women have no wish to be taken back, leaving to their countrymen no option except their capture. The Romans captors had treated them well, they had married them and made them bear children. Titus Livius identifies the radical change in the women's feelings, and narrates how the seemingly inevitable confrontation ended instead with the reconciliation of the two parties [Li]:

> Then it was that the Sabine women, whose wrongs had led to the war, throwing off all womanish fears in their distress, went boldly into the midst of the flying missiles with dishevelled hair and rent garments. Running across the space between the two armies they tried to stop any further fighting and calm the excited passions by appealing to their fathers in the one army and their husbands in the other not to bring upon themselves a curse by staining their hands with the blood of a father-in-law or a son-in-law, nor upon their posterity the taint of parricide. "If," they cried, " you are weary of these ties of kindred, these marriage-bonds, then turn your anger upon us; it is we who are the cause of the war, it is we who have wounded and slain our husbands and fathers. Better for us to perish rather than live without one or the other of you, as widows or as orphans."

In contrast, modern history provides some distinctly regrettable examples of (c), categorized by psychiatrist Nils Bejerot as the *Stockholm syndrome*; one case in point is the abduction by a group of terrorists of Patricia Hearst, daughter of a millionaire, who ended up joining her tormentors in the practice of crimes, and was captured by the police in an apartment at San Francisco [HM].

The legendary story of *Helen of Troy*, in spite of various discordant interpretations, seems to be a remarkable example of (d). Married to king Menelaus of Sparta, Helen fled to Troy in the company of Paris, out of her free will according to a number of versions. In his *Heroides*, Ovid elaborates on the subtle work of seduction performed by Paris, and on Helen's gradual surrender to this man, a protégé of Aphrodite (Venus), goddess of love. Helen is shown writing a response to Paris [Ov]:

> As for your loud vaunting and talk of brave deeds, that face belies your words. Your parts are better suited for Venus than for Mars. Be the waging of wars for the valiant; for you, Paris, ever to love. Bid Hector, whom you praise, go warring in your stead; 'tis the other campaigning befits your prowess. That prowess, were I wise or something bolder, I would employ; employed it will be by whatever maid is wise – or I perchance, forgetting modesty, shall learn wisdom and, overcome by time, yield in tardy surrender.

But, after their escapade to Troy, where they get married, her love feelings start to wane while the Trojan war follows its bloody course, as she recalls the far manlier Menelaus. Homer signals repeatedly this critical change of sentiment in the *Iliad* [Ho]:

> There Helen, daughter of Zeus who holds the aegis, took her seat, turning her eyes aside, and spoke slightly to her husband: "You come back from the fighting, then. I wish you had died there, brought down by a man of strength, who was once my husband."

Not surprisingly, her recovery by Menelaus turned from capture to rescue, as Virgil registers in the *Aeneid*. Paris was dead, and she had been delivered to Paris's brother Deiphobus. When the Greeks came out of the wooden horse and stormed the Trojan palaces, Helen herself made sure that Menelaus should win – and know that she was helping him in atonement for her previous misconduct. The shadow of Deiphobus reports the episode to Aeneas; and what better example of irony could we find than his calling Helen "this peerless wife"? [Vi]:

> Meanwhile, this peerless wife takes every weapon from the house – even from under my head she had withdrawn my trusty sword; into the house she calls Menelaus and flings wide the door, hoping, I doubt not, that her lover would find herein a great boon, and so the fame of old misdeeds might be blotted out.

One more example of (d) appears in the story of *Tristan and Isolde* in Béroul's version [Ber]. The knight had eloped with the queen, they were living in harsh conditions in a forest. The dramatic change of their love feelings, which allowed Yseut's rescue by king Mark to be operated by a simple invitation, with no need to fight, had a very curious cause – the timely expiry date of the love potion they had drunk a few years before, when sailing from Ireland to Cornwall [Ber]:

> The day after St. John's day the three years to which the potion was limited came to an end . . . Hear now how Yseut felt. She said to herself: "Alas, poor wretch, why were you given youth? You are

here in the wood like a slave, you can find few people to serve you. I am a queen, but I lost that name through the potion which we drank on the sea . . . I ought to have around me in my rooms the damsels of the kingdom and the daughters of the free vassals . . . Tristan, the person who brought us the love potion to drink led us sadly astray; we could not have been more deceived."

Before we pass to technical matters, it is necessary to point out some limitations on what we propose to attempt in a computer environment. Literary texts present complexities that cannot be easily dealt with. The *Ramayana* is a case in point. On first glance it may appear a conventional abduction-rescue narrative, with a love triangle and clear-cut goals of the hero and the villain, fighting each other for the possession of the princess. But the villainy and the hero's determination to retaliate were just a pretense machinated by the gods, whose goal was the destruction of the *rakshasa* (a demon-like being) Ravana, condemned by them as a trouble-maker. Having given him extraordinary powers, of which he abused with the utmost insolence, and the promise that no supernatural entity would ever harm him, they had to resort to an extreme solution: the god Vishnu incarnated as a man, prince Rama himself, without the memory however that he was an *avatar* of the god rather than a mere mortal. As a man, Rama was not bound to the promise of not attacking Ravana, which he did for the (apparent) purpose of recovering Sita.

And over and above the intentions of the *devas* (divine beings) was the over-arching goal of Valmiki, the author, to produce a book for the edification of whoever managed to read even a little portion of it. In a long epilogue, Valmiki enumerates the blessings, both spiritual and material, that would reward his devoted readers (ourselves included, hopefully).

As one might anticipate, we shall not try to cope with such complicated goal structures. Yet it is important to recognize the need, if one eventually may want to upgrade a basic set of facilities like ours towards a full-fledged storytelling system, that a plurality of goal levels must, at some point, be considered. The description of the **Minstrel** system [Tu], for instance, postulates four classes of goals that a conscientious author should be concerned with, namely: thematic, consistency, drama and presentation goals.

Moreover we shall only consider linear plots, leaving out therefore the multi-level narratives, so common in Indian or Arabian literature, in the style of the *One Thousand and One Nights* [Ben]. Incidentally, the *Ramayana* has a still more convoluted structure. Close to the end, Rama condemns the virtuous Sita to an undeserved exile, where their twin children are born, and raised in Valmiki's company. At one time, they go to Rama's court, where they start to recite for their father a narrative they had learned from Valmiki – the *Ramayana* itself – which makes the Sanskrit epic recursive!

4. A basic level prototype

A Prolog prototype was developed to experiment, naturally on a modest scale, with the notions expounded here. The main engine behind the implementation is an extension of *Warplan*, a backward-chaining plan-generator produced in 1974 by David Warren [Wa].

4.1. Conceptual schemas

We have been working with a conceptual design method involving three schemas: static, dynamic and behavioural.

The *static schema* specifies, in terms of the *Entity-Relationship* model [BCN], what are the entity and relationship classes and their attributes. In our simple example, character and place are entities. The attributes of characters are name, which serves as identifier, and gender. Places have only one identifying attribute, pname. Characters are pairwise related by relationships loves, held_by and consents_with. The last two can only hold between a female and a male character; held_by(Sita,Ravana) means that Sita is forcefully constrained by Ravana, whereas consents_with(Sita,Ravana) would indicate that Sita has voluntarily accepted Ravana's company. Two relationships associate characters with places: home and current_place.

The *dynamic schema* defines the operations that will consistently change the state of the world, by altering the properties of entity instances. The *STRIPS* [FN] model is used. Each operation is defined in terms of pre-conditions, which consist of conjunctions of positive or negative literals, and any number of post-conditions, consisting of facts to be asserted or retracted as the effect of executing the operation.

We have provided operations at two levels. The four main events are performed by level-1 operations: abduct, elope, rescue and capture. Operations at level-2 are actions of smaller granularity, in terms of which the level-1 operations can be detailed: ride, entreat, seize, defeat, and carry.

Between level-1 and level-2 operations holds therefore a meronymic relation, which however is not rigidly determined, i.e. the correspondence between a level-1 operation and a sequence of level-2 operations is, so to speak, context-sensitive, depending on the current state. For instance, abduct(Rama,Sita) can be expressed by seize(Ravana,Sita) followed by carry(Ravana,Sita,Lanka) if both the victim and the ravisher are currently at the same place, but will need a preliminary ride(Ravana,Lanka,forest) if Sita is in the forest and Ravana in his home at Lanka.

Our present version of the *behavioural schema* consists of goal-inference (also called situation/objective) rules, belief rules, and emotional condition rules.

For the example, three goal-inference rules are supplied. The first one refers to the ravisher. In words, in a situation where the princess is not in her home and the hero is not in her company – and hence she is unprotected – the ravisher will want to do whatever is adequate to bring her to his home. The other goal-inference rules refer to the hero, in two different situations having in common the fact that the ravisher has the woman in his home: either the hero believes that she does not love the other man, or he believes that she does. In both situations, he will want to bring her back, freely in the first case and constrained in the second.

The belief rules are somewhat rational, but notice that they are treated as defaults, which can be overruled as will be described in section 4.5. A man (the hero or the ravisher) believes that the woman does *not* love his rival if the latter has her confined, but if she has ever been observed in his company and in no occasion (state) was physically constrained, the conclusion will be that she is consenting (an attitude that would seem too subjective to be ascertained directly in a realistic context).

The emotional condition rules refer to the three characters. A man (or woman) is happy if currently in the company of his (or her) beloved, and bored otherwise. A special condition applies to the woman: she will be *absolutely* happy if, in addition to the first motive for contentment, she has never been constrained by any of the two adversaries. Is this most felicitous outcome feasible? As we shall see it can happen in two rather curious cases, whose description we postpone to the end of section 4.5.

4.2 Main features of the plan-generator

The plan generator is activated by calling plans(O, P), where O is the given objetive and P the plan to be generated. O can take the form of a single positive – F – or negative – not F – literal denoting a fact of a predefined class, but can also be a conjunction involving such literals and arbitrary expressions, such as comparisons. Take for example:

```
?- plans((current_place('Sita',L1), home('Sita',L2), not (L1 == L2)), P).

P = start
```

expressing that, at the state to be reached by executing plan P, Sita should be at a place different from her home. If the objective is already true at the current state, as happens at the initial state of our example (wherein Sita is in the forest), P is reduced to the start symbol. In addition, if the plan generator is caused to backtrack, P may receive alternative instantiations. By default, set by the initial null value of a controlling clause, op_level(0), operations of different levels may coexist in P; if only level-2 operations are wanted, one should first issue the command

```
:- set(op_level(2)).
```

in which case one result of the above call to plans will be:

```
P = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)
```

As this arrow-based notation suggests, plans are represented by totally ordered sequences, beginning with start, and formed by terms expressing the execution of operations defined at the dynamic schema. But notice that, since the order imposed by the pre- and post-conditions specified in the dynamic schema is only partial in general, backtracking can also produce alternatives containing exactly the same operations, the only difference being their position in the plan.

The plan generator follows a backward chaining strategy. For a fact F (or not F) that is part of the given objective, it checks whether it is already true (or false) at the current state. If it is not, it looks for an operation Op declared to add (or delete) the fact as part of its effects. Having found such operation, it then checks whether the pre-condition Pr of Op currently holds – if not, it proceeds recursively trying to satisfy Pr. In addition, the plan generator must consider the so-called frame problem [Lo], by establishing (in a second-order logic notation) that, if a pre-condition Pr was satisfied either at the initial or at some intermediate state reached by the plan, and some subsequent Op' operation is included in the plan for which no deleted (or added) effect that might invalidate Pr was explicitly declared, then Pr stays valid.

Like objectives, pre-conditions are denoted by conjunctions of literals and arbitrary logical expressions. We distinguish, and treat differently, three cases for the occurrence of positive or negative facts in pre-conditions:

a. facts which, in case of failure, should be treated as objectives to be achieved recursively by the plan generator;
b. facts to be tested immediately before the execution of the operation, but which will not be treated as objectives in case of failure: if they fail the operation simply cannot be applied;
c. facts that are not declared as added or deleted by any of the pre-defined operations.

Recall that the general format of a pre-condition clause is precond(Op, Pr) :- B. In cases (a) and (b), a fact F (or not F) must figure in Pr, with the distinction that the barred notation /F (or /(not F)) will be used in case (b). Case (c) is handled in a particularly efficient way. Since it refers to facts that are invariant with respect to the operations, such facts are included in the body B of the clause, being tested a single time against the current state when the clause is selected.

Take, for instance, the precond clause of operation seize(M,W), where M is the agent and W the patient of the action. Clearly the two characters should be together at the same place, and accordingly the Pr argument shows two terms containing the same variable P to express this requirement, but the term corresponding to W is barred: /current_place(W,P), which does not happen in M's case. The difference has an intuitive justification: the prospective agent has to go to the place where the patient is, but the latter will just happen to be there for some other reason.

The proper treatment of (a) and (b) is somewhat tricky. Suppose the pre-condition Pr of operation Op is tested at a state T1. If it fails, the terms belonging to case (a) will cause a recursive call whereby one or more additional operations will be inserted so as to move from T1 to a state T2 where Op itself can be included. It is only at T2, not at T1, that the barred terms in case (b) ought to be tested, and so the test must be delayed until the return from the recursive call, when the plan sequence able to reach T2 will be fully instantiated.

Operations can admit more than one precond cause, so as to cope with different circumstances. This happens with the carry(M,W,P2) operation, whereby W will either freely consent to be transported to P2 by M, or will have to be forcefully held by him.

Regarding the added and deleted clauses declaring effects of operations, the plan generator also employs a barred notation, to distinguish two cases: a. primary effects, b. secondary unessential effects. If any fact F in case (a) to be added by Op already holds, or already does not hold if the should be deleted, then Op is considered *non-productive* and fails to be included in the plan. In contrast, in case (b), such lack of effect would be admitted and therefore would not cause failure.

As an example, consider the clause of operation capture(M1,W) that declares as deleted the fact held_by(W,M2), as a result of M1's action to take away W from M2. Notice that the fact may or may not hold prior to capture; it will hold if W was abducted by M2, but will not hold if an elopement occurred instead – and that is why the barred notation is used for this particular deleted clause. On the contrary, the fact current_place(W,P2), where P2 is the home of M2, must necessarily be deleted by an effective execution of the operation, and so does not figure as barred.

Plans, either manually composed by the user or automatically generated (with or without the user intervention), can be executed by calling execute(P), which performs assert or retract commands on the facts to be, respectively, added or deleted. But execute always checks the plan's pre- and post-conditions, failing and causing no effect if anything is wrong. A log(L) literal, initiated with L=start, is extended with each successful plan execution and can be usefully retrieved for a variety of purposes. On the basis of the log and of the initial state, which is saved when a session begins, it is possible to query about facts at any intermediate state. It is also possible to restore a previous state S (initial or intermediate) that has been saved, which enables simulation runs.

User interventions, necessary to achieve unplanned situations, are permitted in a limited scale through *directives* that can be either intermixed with the operations in a plan or called separately. Two of these are illustrated in section 4.5, one for changing loves facts, immune to the predefined operations, and the characters' beliefs, which may or may not correspond to the actual facts.

To finish this partial review of the plan features, we remark that plans(O,P) can be called in more than one way. More frequently O is given, as the objective, and P is a variable to which a generated plan will be assigned as output. However an inverse usage has been provided, wherein P is given and O is a variable; in this case, the algorithm will check whether P is valid and, if so, assign its net effects (a conjunction of F and not F terms) to O.

4.3. Coherent sequences

Moving along the *syntagmatic* axis (fig. 1) is primarily the task of the plan-generator, as it composes a coherent plot by aligning events in view of the pre- and post-conditions of the appropriate predefined operations.

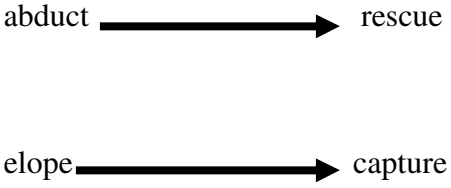

Fig. 1. syntagmatic relations

With an initial state situation wherein Sita loves Rama, the plot below is generated to end up with Sita in the palace. Notice that the arrow-based notation is translated into pseudo-natural language, through the application of a simple-minded template device.

```
?- plans(current_place('Sita',palace),P), narrate(P).

Ravana abducts Sita. Rama rescues Sita.

P = start=>abduct(Ravana,Sita)=>rescue(Rama,Sita)
```

The execute command applies to both single operations and plans. As a side-effect, it updates the log as shown below.

```
?- execute(abduct('Ravana','Sita')).
```

?- log(L).

L = start=>abduct(Ravana, Sita)

Instead of indicating the objective to be pursued by the plan generator, one may leave the choice to the existing goal-inference rules. If rule_plan(C,S,O,P) is called, it will try to find a rule sit_obj(C,S,O) such that S is currently true, thus inducing O as an objective for a character C; in case of success, O is passed internally to plans(O,P).

?- rule_plan(C,S,O,P), narrate(P).

Rama rescues Sita.

C = Rama
S = current_place(Sita, Lanka), believes(Rama, not loves(Sita, Ravana))
O = current_place(Sita, palace), not held_by(Sita, Rama)
P = start=>rescue(Rama, Sita)

A user wanting to generate a plan including certain specific operations, aligned in an incomplete Pi plan sequence, can ask the plan generator to fill it up to a complete and valid plan Po, adequate for reaching some objective O. If no objective is specified, the plan generator will assume that it corresponds to the effects of the last operation supplied, but of course all operations in Pi have to be part of the generated Po, and so their intermediate effects will be contemplated.

?- fill_up(O,start=>carry('Ravana','Sita','Lanka')=>carry('Rama','Sita',palace),Po),narrate(Po).

Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka. Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.

O = current_place(Sita, palace), not current_place(Sita, forest)
Po = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)

As mentioned earlier, plans are in general only partially ordered. Whenever this is the case, after the plan generator has produced a plan P, it will display if caused to backtrack other plans with the same operations contained in P, differing only in their sequence. A call to plan_seq(P,Ps) offers the user a controlled way to obtain a preferred sequence. Taking as input the original P, this facility will produce a reordered plan Ps, prompting the user to indicate a choice if more than one operation is ready to be included (i.e. has its pre-conditions satisfied). In the example below, the objective given to the plan generator asks for both forms of submission of the patient to the agent's will: consents_with and held_by. The plan generator responds with a plan P wherein operations entreat and seize achieve, respectively, these two effects. Curiously, both the original P and the reordered plan Ps, accomodated to the user's indications, suggest stories that may well happen in reality or fiction. In P, a voluntary elopement is disguised as an abduction, whereas Ps can be interpreted as an overt elopement after which the seducer decides to restrict the woman's freedom.

?- plans((consents_with('Sita', 'Ravana'), held_by('Sita', 'Ravana'), not current_place('Sita', forest)), P), plan_seq(P,Ps), narrate(P), narrate(Ps).

[***f2:entreat(Ravana, Sita), f3:seize(Ravana, Sita)***]

choose: f2.
[*f3:seize(Ravana, Sita), f4:carry(Ravana, Sita, Lanka)*]
choose: f4.

Ravana rides from Lanka to forest. Ravana entreats Sita. Ravana seizes Sita. Ravana carries Sita to Lanka.

Ravana rides from Lanka to forest. Ravana entreats Sita. Ravana carries Sita to Lanka. Ravana seizes Sita.

P = start=>ride(Ravana, Lanka, forest)=>entreat(Ravana, Sita)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)
Ps = start=>ride(Ravana, Lanka, forest)=>entreat(Ravana, Sita)=>carry(Ravana, Sita, Lanka)=>seize(Ravana, Sita)

The algorithm utilized by plan_seq(P,Ps) first converts P into a particularly convenient plot format Pf, where operations are shown in a list prefixed with tags, and tag pairs in a second list express the ordering requirements induced by the pre- post-condintions dependencies. A tag pair fi-fj indicates that the operation with tag ti must be executed before the operation with tag tj. To visualize Pf, one may call plot_format(P,Pf).

?- plans((consents_with('Sita', 'Ravana'), held_by('Sita', 'Ravana'), not current_place('Sita', forest)),P), plot_format(P,Pf).

Pf = [[f1:ride(Ravana, Lanka, forest), f2:entreat(Ravana, Sita), f3:seize(Ravana, Sita), f4:carry(Ravana, Sita, Lanka)], [f1-f2, f1-f3, f2-f4, f3-f4]]

4.4. Alternative choices

Moving along the *paradigmatic* axis (fig. 2) gives ampler opportunity to obtain different plots than simply choosing an appropiate sequence among those conforming to the partial order requirements.

Fig. 2. paradigmatic relations

Still by simple backtracking, with the call to plans(O,P) and the initial state described at the beginning of the previous section, a plot involving capture instead of rescue will be provided.

?- plans(current_place('Sita',palace),P), narrate(P).

Ravana abducts Sita. Rama captures Sita.

If at the initial state Sita loves Ravana instead of Rama, the former has two ways to bring the princess to his dwelling, whereas the latter can only force her to return.

?- forall(plans(current_place('Sita',palace),P), narrate(P)).

Ravana abducts Sita. Rama captures Sita.
Ravana elopes with Sita. Rama captures Sita.

Resorting to violence, as in abduction or capture, can be excessive and unnecessary when the patient of the action loves the agent, but our example specification allows that. Better motivated plots take into consideration what objectives may move the characters as they appraise the facts at the current situation, as also what they rightly or wrongly believe about them. Let us start again from a state wherein Sita loves Rama. Calling may_wish(C,S,O), whereby the goal inference rules of the behavioural schema are activated, only one answer is supplied, involving Ravana: Sita is alone in the forest, vulnerable therefore to him. Since she does not love the stranger, the plan generator leaves him just one possibility: carrying her by force. Using this time the level-2 operators, the tasks of determining the objective O of character C, finding a plan P for O, and executing P are achieved by entering the line:

?- may_wish(C,S,O), plans(O,P), execute(P), narrate(P).

Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka.

C = Ravana
S = current_place(Sita, forest), not current_place(Rama, forest)
O = current_place(Sita, Lanka)
P = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)

At the new state, we can submit again the same line, obtaining a single answer which now refers to Rama. He wants to save Sita and, from the fact that she is held by force, concludes, through the application of the second belief rule specified at the behavioural schema, that she does not love the ravisher – which in turn determines Rama's preference for not constraining her, expressed by the negated literal not held_by(Rama, Sita) included in objective O.

?- may_wish(C,S,O), plans(O,P), execute(P), narrate(P).

Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.

C = Rama
S = current_place(Sita, Lanka), believes(Rama, not loves(Sita, Ravana))
O = current_place(Sita, palace), not held_by(Sita, Rama)
P = start=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)

Since both plans were executed they are now part of the log, which can be inspected later if one wants to retrieve any subsequences that may correspond to motivated plans. The algorithm behind observed_plan(C,S,O,P) moves forward along the log searching for some state **s1** where the situation S of a sit_obj(C,S,O) goal-inference rule holds. Finding such state **s1**, it advances until a state **s2** is found wherein O holds. It then assumes that the subsequence of operations between **s1** and **s2** contains a plan adequate to bring about the desired state transition, which can be extracted from the subsequence by a filtering process. Filtering is done by moving backward from the last operation in the subsequence, and eliminating any operation that neither contributes to O nor to the pre-conditions of the operations thus far retained. In our example, filtering is not necessary but, in general, a log

may register identifiable plans interspersed with any number of operations executed for other unrelated purposes.

?- observed_plan('Ravana', S, O, P_Ravana), observed_plan('Rama', S1, O1, P_Rama), narrate(P_Ravana), narrate(P_Rama).

Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka.

Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.

S = current_place(Sita, forest), not current_place(Rama, forest)
O = current_place(Sita, Lanka)
P_Ravana = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)
S1 = current_place(Sita, Lanka), believes(Rama, not loves(Sita, Ravana))
O1 = current_place(Sita, palace), not held_by(Sita, Rama)
P_Rama = start=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)

When looking for the **s1** and **s2** states, the algorithm has to recapitulate the execution of the operations in the log, starting with the initial state saved at the beginning of the session. The same facility can also be applied to a long given sequence Sq of operations, instead of working on the log, to find in it one or more plans. For this purpose, the sequence Sq to be examined must be supplied as an additional parameter: observed_plan(C,S,O,Sq,P). When working on such still non-executed sequences, the current state thus far reached is of course used as initial.

4.5. Shifts along the way

Until this point we restricted ourselves to planned and hence well-behaved plots. It is time now to introduce a measure of transgression, disrupting the context so as to allow the composition of plots containing events in *antithetic* relation (fig. 3).

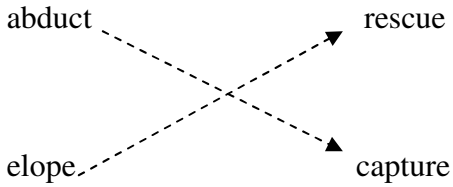

Fig. 3. antithetic relations

The user, as "deus ex machina", can interfere with the plan generation discipline by issuing *directives*, which may be called independently or else inserted in plans.

One directive is make_believe(C, B), arbitrarily assigning a belief B to character C, which overrules any previous belief on the same facts, either specified through the belief rules of the behavioural schema or stated by a previous application of the make_believe directive itself. Coming back to the example of the preceding section, suppose the user adds a directive at the end of the plan described before, whereby Ravana seizes Sita and compels her to go with him to Lanka. The directive, in a manner quite congenial to Shakespeare's Iago, makes Rama believe, against the evidence supplied by the applicable belief rule, that

Sita loves the ravisher. The operations proper are the same as before, with the same effect on the factual context; the difference has to do with Rama's beliefs.

```
?- may_wish(C, S, O), plans(O, P), add_after(make_believe('Rama', loves('Sita', 'Ravana')),P,P1), execute1(P1), beliefs.

   Rama believes that Sita loves Ravana

C = Ravana
S = current_place(Sita, forest), not current_place(Rama, forest)
O = current_place(Sita, Lanka)
P = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)
P1 = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita,
Lanka)=>make_believe(Rama, loves(Sita, Ravana))
```

As a consequence of this induced belief, the line that we have been considering yields a plot wherein, ironically, the hero forces the princess to return to their home, exactly as she wanted.

```
?- may_wish(C,S,O), plans(O,P), execute(P), narrate(P).

Rama rides from palace to Lanka. Rama defeats Ravana. Rama seizes Sita. Rama carries Sita to palace

C = Rama
S = current_place(Sita, Lanka), believes(Rama, loves(Sita, Ravana))
O = current_place(Sita, palace), held_by(Sita, Rama)
P = start=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>seize(Rama, Sita)=>carry(Rama, Sita,
palace)
```

Suppose now that the user manually composes the plot

```
P = start=>elope('Ravana','Sita')=>rescue('Rama','Sita')
```

that, already in section 2.3, we described as unacceptable in principle, since it consists of two events that presuppose mutually incompatible contexts. And this time it is not just a matter of incorrect beliefs, since the conflict occurs at the factual level: the pre-conditions of elope and rescue diverge with respect to who is the object of the woman's love. She should willingly escape with the seducer, and afterwards – even more ironically – should expect to be saved from him, just like Helen in Virgil's version of the legend.

Assuming an initial state wherein Sita loves Rama, not Ravana, her feelings must change before elopement, and then, again, before delivering herself to the rescuer. The user is prompted (as shown in boldface, italic) to determine "ex-machina" these changes, interacting with the system to indicate what facts should vary in their truth values. An expression of the form F:C can be used to reply to the query, being read by the system as referring to all instances of fact F satisfying an arbitrary condition C . In the example, the reply refers to Sita's feelings for each of the two male character in the story. Notice the presence of the vary directive, inserted twice in P1.

```
?- P = start=>elope('Ravana','Sita')=>rescue('Rama','Sita'), tamper(P,P1), execute1(P1).

elope(Ravana, Sita) not applicable
  vary context?   loves('Sita',M):gender(M,male).
rescue(Rama, Sita) not applicable
  vary context?   loves('Sita',M):gender(M,male).
```

P = start=>elope(Ravana, Sita)=>rescue(Rama, Sita)
P1 = start=>vary(loves(Sita, M):gender(M, male))=>elope(Ravana, Sita)=>vary(loves(Sita, M):gender(M, male))=>rescue(Rama, Sita)

Although not so uncommon, this is possibly the most intriguing plot that we have considered in the limited universe of our example, due to the capricious reversal in the middle of the story. And, curiously, this is the only one thus far whose outcome is completely satisfactory to the female character: she ends up with the man that she (currently) loves, and in no intermediate state is ever constrained. This can be verified, after the plot is executed and logged, by entering:

?- log(L), describe(emotional_condition('Sita',C,L)).

Sita is absolutely happy

L = start=>elope(Ravana, Sita)=>rescue(Rama, Sita)
C = absolutely_happy

And yet we found later, with some surprise, another plot having an equally happy end. We stumbled upon it by explicitly requesting the desired outcome. What makes the obtained plot the quintessence of irony is that it tells a non-story... There is no villainy and hence no retaliation, none of the four events affecting the love triangle takes place. Rama simply joins the lovely Sita in the forest and carries her back to their princely home.

?- plans(current_place('Sita', L), P), emotional_condition('Sita', absolutely_happy, P), narrate(P), describe(emotional_condition('Sita', C, P)).

Rama rides from palace to forest. Rama entreats Sita. Rama carries Sita to palace.

Sita is absolutely happy

L = palace
P = start=>ride(Rama, palace, forest)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)
C = absolutely_happy

Exclusively for experimenting with the present example, we provided two ad-hoc versions of the vary directive, having in mind the status of the loves relationship from the part of the victim W. Three cases are considered: (0) she loves no one; (1) she loves the hero; (3) she loves the ravisher. One version, vary(W), makes a random choice: it reads the current time T registered by the system's clock and obtains 0, 1 or 2 by calculating T mod 3. The other version, vary(W,I), allows to indicate the chosen number explicitly in the second parameter; the choice of a zero value recognizes the two male characters as the only constant lovers.

?- gender(W,female), vary(W,0).

?- forall(loves(C1,C2), describe(loves(C1,C2))).

Rama loves Sita

Ravana loves Sita

4.6. Down to details

Until now we have been alternating level-1 and level-2 operations in the examples. Between the two levels there are *meronymic* relations (fig. 4) that will now be exploited. Creating plots in hierarchic fashion is a most common practice, starting with a broad view of the events, which in the case of our example corresponds to the level-1 operators.

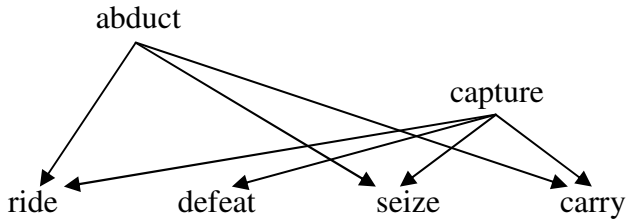

Fig. 4.1. meronymic relations - the forceful actions

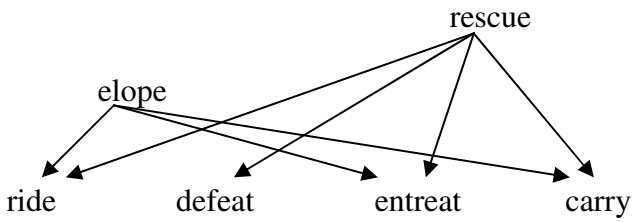

Fig. 4.2. meronymic relations - the gentle actions

At later stages, one would gradually decompose each event into finer grain actions, eventually to the point of coordinated physical movements, as required for displaying animated scenes [CPFF]. The decomposition of each level-1 operation Op into level-2 operations is treated as a plan generation task, started at the state immediately before the application of Op, which should correspond to a *situation* wherein the pre-conditions of the operation should hold, and taking the effects of Op as *goal* to be reached by the resulting plan to be formed from the repertoire level-2 operations. More than one such result may be possible, depending on the initial state and on the changes effected by the preceding operations. Our detail facility asks the user, for each input level-1 operation, whether the decomposition it has produced is considered acceptable.

A particularly simple decomposition is obtained for the routine abduction-rescue plot.

?- set_op_level(1), plans(current_place('Sita',palace),P), detail(P,Pd), narrate(P), narrate(Pd).

Detailed version for: abduct(Ravana, Sita)
start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)
Acceptable?(yes/no) yes.

Detailed version for: rescue(Rama, Sita)
start=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)
Acceptable?(yes/no) yes.

Ravana abducts Sita. Rama rescues Sita.

Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka. Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.

P = start=>abduct(Ravana, Sita)=>rescue(Rama, Sita)
Pd = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)

The inverse of detailing, summarizing, may also be useful. Our prototype supplies a rather limited version, which only works if the detailed plan is divisible into subsequences that can be exactly identified with level-1 operations.This means that the process fails if other extraneous operations intervene. In other words, summarize(P1,P2) succeeds if and only if detail(P2,P1) also does. As expected, then, the following line reverts the result of the previous run.

?- Pd = start=>ride('Ravana', 'Lanka', forest)=>seize('Ravana', 'Sita')=>carry('Ravana', 'Sita', 'Lanka')=>ride('Rama', palace, 'Lanka')=>defeat('Rama', 'Ravana')=>entreat('Rama', 'Sita')=>carry('Rama', 'Sita', palace), summarize(Pd, Ps), narrate(Pd), narrate(Ps).

Ravana rides from Lanka to forest. Ravana seizes Sita. Ravana carries Sita to Lanka. Rama rides from palace to Lanka. Rama defeats Ravana. Rama entreats Sita. Rama carries Sita to palace.

Ravana abducts Sita. Rama rescues Sita.

Pd = start=>ride(Ravana, Lanka, forest)=>seize(Ravana, Sita)=>carry(Ravana, Sita, Lanka)=>ride(Rama, palace, Lanka)=>defeat(Rama, Ravana)=>entreat(Rama, Sita)=>carry(Rama, Sita, palace)
Ps = [start=>abduct(Ravana, Sita)=>rescue(Rama, Sita)]

A little more flexibility is afforded by summarize1(O,P,Ls), which, using a given objective O, calls internally the fill_up facility introduced at section 4.3, extends the possibly incomplete level-2 plan sequence P in all alternative ways adequate to achieve O, summarizes each of them, and then places the obtained level-1 plans in list Ls. The overall effect is comparable to observed_plan (cf. section 4.4), being therefore one more way of *recognizing*, from a few observed actions, what a character are trying to do and for what objective. For instance, if one notices that Ravana is riding to the forest, what could he be attempting, assuming that he is loved by the princess? Well, knowing what is his pressing desire, he can be either intent on elopement or abduction (the detailed level-2 plans generated by fill_up are not exhibited, since they are consumed internally).

?- vary('Sita',2), P = start=>ride('Ravana','Lanka',forest), may_wish('Ravana',S,O), summarize1(O,P,Ps), forall(on(Pi,Ps), narrate(Pi)).

Ravana abducts Sita.

Ravana elopes with Sita.

P = start=>ride(Ravana, Lanka, forest)
S = current_place(Sita, forest), not current_place(Rama, forest)
O = current_place(Sita, Lanka)
Pr = [start=>abduct(Ravana, Sita), start=>elope(Ravana, Sita)]

Plan generation is surely more directly relevant to the composition and adaptation of plots than the recognition of plans and objectives. But the latter task looks very convenient if the set of facilities described in this paper is used in an interactive plan-supported game-

playing context, since each player can employ automatic recognition as an aid to discover what the opponents are trying to do.

5. Concluding remarks

Although the process of plot composition could surely be enriched far beyond what was presented here, our four-sided approach seems to offer a sound basic groundwork for future developments. As argued in section 2.3, the conjecture that the interplay of the syntagmatic, paradigmatic, antithetic and meronymic relations already permits an ample coverage is reinforced by the connection established between these relations and the four major tropes.

Our proposed facilities can be combined for several kinds of uses, under suitable user interfaces. To begin with, they can operate in environments similar to that of our **Logtell** plot composition and animation system [CPFF]. An even more straightforward application would be a storyboard tool [THA], the construction of which is being considered within our project.

In addition, the facilties can be usefully incorporated into larger systems, either in the digital entertainment area, including game-playing, or for simulation and training in the area of business information systems.

Finally, let us recall that we have addressed the *fabula* level only, where one simply indicates *which* events should be included in the plots. One especially complex problem to be faced at the next level – the *story* level, where the concern is *how* to tell the events – is to find an adequate justification for the contextual disruptions, in our case introduced *ex machina* via user interaction. Such elaborations may be immediately plausible, like Helen's gradual disillusion with the martial virtues of Paris, or may appeal to knowledge made popular by the media, like the Stockolm syndrome to explain the victim's conversion to the terrorists' ideology, or even be fanciful, like the expiration of the love potion in the *Tristan* romance. If we see a disruption not as a discontinuity in one context, but as an attempt to put together two originally incompatible contexts, then the notion of *blending* [FT] immediately comes to mind, as the technique or artisanship of conciliating the pending conflicts, which often requires a great deal of creativity.

At the third and last level – the *text* level – the narrative is represented in some medium, not necessarily printed pages. We have done some initial work [FC] on the generation of natural language texts from plots of log-registered transactions, noting that the result should still be enhanced through the application of methods pertaining to computational linguistics [Mk]. As shown in [CPFF], computer graphic animation provides a particularly attractive way for displaying narrative plots, both in the realm of literary genres or of business information systems.

References

[Ar] Aristotle. "Poetics". In *Classical Literary Criticism*. Penelope Murray et al (trans.). London: Penguin, 2000.

[BBFC] Simone D. J. Barbosa, Karin K. Breitman, Antonio L. Furtado and Marco A. Casanova. "Similarity and Analogy over Application Domains". In *Proc. of XXII Simpósio Brasileiro de Banco de Dados*, 2007.

[BCN] C. Batini, S. Ceri, S. and Navathe. *Conceptual Design – an Entity-Relationship Approach*. Benjamin Cummings, 1992.

[BCT] Karin K. Breitman, Marco A. Casanova and Walter Truszkowski. S*emantic Web: Concepts, Technologies and Applications*. London: Sringer, 2007.
[Ben] J. E. Bencheikh (trans.). *Les Mille et Une Nuits*. Paris: Gallimard, 2006.
[Ber] Béroul. *The Romance of Tristan*. A. S. Fedrick (trans.). London: Penguin, 1970.
[Bo] W. Booth. *A rhetoric of Tropes*. Chicago: University of Chicago Press, 1974.
[Bu] Kenneth Burke. *A Grammar of Motives*. Los Angeles: University of California Press, 1969.
[Cha] Daniel Chandler. *Semiotics: The Basics*. London: Routledge, 2007.
[Chr] Chrétien de Troyes. *Le Chevalier de la Charrete*. Mario Rocques (ed.). Paris: Honoré Champion, 1983.
[CPFF] Angelo E. M. Ciarlini, Cesar T. Pozzer, Antonio L. Furtado and Bruno Feijó. "A logic-based tool for interactive generation and dramatization of stories". In Proc. of *Advances in Computer Entertainment Technology*, Valencia, Spain, 2005.
[Cu] Jonathan Culler. *On Deconstruction: Theory and Criticism after Structuralism*. Ithaca: Cornell University Press, 1983.
[FC] Antonio L. Furtado and Angelo E. M. Ciarlini. "Generating Narratives from Plots using Schema Information". In Proc. of the *5th International Conference on Applications of Natural Language to Information Systems*, Versailles, France, 2000.
[FN] R. E. Fikes, and N. J. Nilsson. "STRIPS: A new approach to the application of theorem proving to problem solving". *Artificial Intelligence* , 2(3-4), 1971.
[FT] G. Fauconnier and M. Turner. *The Way We Think*. New York: Basic Books, 2002.
[FV] Antonio L. Furtado and Paulo A. S. Veloso. "Folklore and Myth in *The Knight of the Cart*". In *Arthuriana*, vol 6, n. 2, pages 28-43, 1996.
[HM] Patricia C. Hearst and Alvin Moscow. *Patty Hearst: her own Story*. New York: Avon, 1988.
[Ho] Homer. *The Odyssey*. E. V. Rieu (trans.). London: Penguin, 1946.
[La] George Lakiff. Women, Fire, and Dangerous Things. Chicago: The University of Chicago Press, 1990.
[Li] Titus Livius. *History of Rome*, vol I. B. O. Fster (trans.). Cambridge: Harvard University Press. Book I, Chapter I: 13 1919.
[LJ] G. Lakoff and M. Johnson. *Metaphors We Live By*. University of Chicago Press, 1980.
[Lo] W. Lloyd. *Foundations of Logic Programming*. Berlin: Springer-Verlag 1987.
[Ma] Christiane Marchello-Nizia (org.). *Tristan et Yseut*. Paris: Gallimard, 1995.
[Mg] Mary McGarry (ed.). "The Story of Deirdre". In *Great Folk Tales of Ireland*. London: Frederick Muller, 1979.
[Mk] K.R. McKeown. *Text Generation: Using Dscourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge: Cambridge University Press, 1992.
[Or] Andrew Ortony (ed.). *Metaphor and Thought*. Cambridge: Cambridge University Press, 1996.
[Ov] Ovid. *Heroides and Amores*. G. Showerman (trans.). Cambridge: Harvard University Press, 1986.
[Pr] V. Propp. *Morphology of the Folktale*. S. Laurence (trans.). Austin: University of Texas Press, 1968.
[RO] P. O'Rorke and A. Ortony. "Explaining Emotions". In *Cognitive Science*, 18 (2) 283-323, 1994.
[Sa] Ferdinand de Saussure. *Cours de Linguistique Générale*. Charles Bally, Albert Sechehaye and Albert Riedlinger (eds.). Paris: Payot, 2006.
[THA] K. N. Truong, G. R. Hayes and G. D. Abowd. "Storyboarding: An Empirical Determination of Best Practices and Effective Guidelines". In Proceedings of the 6th conference on Designing Interactive systems, University Park, 2006.
[Tu] Scott R. Turner. "Minstrel: A computer model of creativity and storytelling". Technical Report UCLA-AI-92-04, Computer Science Department, 1992.
[Va] Valmiki. *Le Ramayana*. Philippe Benoît et al (trans.). Paris: Gallimard, 1999.
[Vi] Virgil. *Eclogues, Georgics, Aeneid*. H. R. Fairclough (trans.). Cambridge: Harvard Univesity Press, 1994.
[Wa] David, H. D. Warren. *A System for Generating Plans*. Edinburgh: University of Edimburgh, Department of Computational Logic, memo 76, June 1974.
[Wh] Hayden White. *Metahistory: The Historical Imagination in Nineteenth-Century Europe*. Baltimore: The Johns Hopkins University Press, 1973.

Appendix A: the prototype tool

A.1. The static schema

```
entity(character,name).
attribute(character,gender).
entity(place,pname).
relationship(loves,[character,character]).
relationship(home,[character,place]).
relationship(current_place,[character,place]).
relationship(held_by,[character,character]).
relationship(consents_with,[character,character]).
```

A.2. The dynamic schema

A.2.1. Level-1 operators

```
operation(abduct(M2,W),1).
deleted(current_place(W,Pc),abduct(M2,W)) :-
  home(M2,P2), not (Pc == P2).
added(current_place(W,P2),abduct(M2,W)) :- home(M2,P2).
added(held_by(W,M2),abduct(M2,W)).
precond(abduct(M2,W),current_place(W,Pc)) :-
   loves(M2,W),
   gender(W,female),
   home(M2,P2),
   home(W,P1),
   not (P1 == P2),
   place(Pc),
   not (Pc == P1),
   not (Pc == P2).

operation(elope(M2,W),1).
deleted(current_place(W,Pc),elope(M2,W)) :-
  home(M2,P2), not (Pc == P2).
added(current_place(W,P2),elope(M2,W)) :- home(M2,P2).
added(consents_with(W,M2),elope(M2,W)).
precond(elope(M2,W),current_place(W,Pc)) :-
   loves(M2,W),
   loves(W,M2),
   gender(W,female),
   home(M2,P2),
   home(W,P1),
   not (P1 == P2),
   place(Pc),
   not (W == M2),
   not (Pc == P1),
   not (Pc == P2).

operation(rescue(M1,W),1).
deleted(current_place(W,P2),rescue(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(current_place(M1,P2),rescue(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(held_by(W,M2),rescue(M1,W)) :-
  home(W,P1), home(M2,P2), not (M2 == M1), not (P2 == P1).
added(current_place(W,P1),rescue(M1,W)) :- home(W,P1).
```

```
/added(current_place(M1,P1),rescue(M1,W)) :- home(W,P1).
added(consents_with(W,M1),rescue(M1,W)).
precond(rescue(M1,W),current_place(W,P2)) :-
   gender(W,female),
   home(W,P1),
   home(M2,P2),
   not loves(W,M2),
   character(M1),
   not (M1 == W),
   not (M1 == M2),
   not (P1 == P2).

operation(capture(M1,W),1).
deleted(current_place(W,P2),capture(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(current_place(M1,P2),capture(M1,W)) :-
  home(W,P1), not (P2 == P1).
/deleted(held_by(W,M2),capture(M1,W)) :-
  home(W,P1), home(M2,P2), not (P2 == P1).
added(held_by(W,M1),capture(M1,W)).
added(current_place(W,P1),capture(M1,W)) :- home(W,P1).
/added(current_place(M1,P1),capture(M1,W)) :- home(W,P1).
precond(capture(M1,W),current_place(W,P2)) :-
   gender(W,female),
   home(W,P1),
   home(M2,P2),
   character(M1),
   not (M1 == W),
   not (M1 == M2),
   not (P1 == P2).
```

A.2.2. Level-2 operators

```
operation(ride(C,P1,P2),2).
deleted(current_place(C,P1),ride(C,P1,P2)) :- not (P1 == P2).
added(current_place(C,P2),ride(C,P1,P2)) :- not (P1 == P2).
precond(ride(C,P1,P2),current_place(W,P2)) :-
  loves(C,W),
  gender(C,male),
  gender(W,female),
  place(P1),
  place(P2),
  not (P1 == P2).

operation(seize(M,W),2).
added(held_by(W,M),seize(M,W)).
precond(seize(M,W),
  (/current_place(W,P),
   current_place(M,P),
   not held_by(W,M2))) :-
  gender(M,male),
  gender(W,female),
  gender(M2,male),
  not (M == M2),
  place(P),
  not home(W,P),
  not home(M,P).
```

```
operation(entreat(M,W),2).
added(consents_with(W,M),entreat(M,W)).
precond(entreat(M,W),
  (/current_place(W,P),
   current_place(M,P),
   /(not held_by(W,M)),
   not held_by(W,M2))) :-
  loves(M,W),
  loves(W,M),
  gender(M,male),
  gender(W,female),
  gender(M2,male),
  not (M == M2),
  place(P),
  not home(M,P),
  not home(W,P).

operation(carry(M,W,P2),2).
deleted(current_place(M,P1),carry(M,W,P2)) :-
  home(M,P2), not (P1 == P2).
deleted(current_place(W,P1),carry(M,W,P2)) :-
  home(M,P2), not (P1 == P2).
added(current_place(M,P2),carry(M,W,P2)) :-
  home(M,P2).
added(current_place(W,P2),carry(M,W,P2)) :-
  home(M,P2).
precond(carry(M,W,P2),
  (consents_with(W,M),
  /current_place(W,P1),
  /current_place(M,P1))) :-
  gender(W,female),
  gender(M,male),
  loves(M,W),
  loves(W,M),
  home(M,P2).
precond(carry(M,W,P2),
  (held_by(W,M),
  /current_place(W,P1),
  /current_place(M,P1))) :-
  gender(W,female),
  gender(M,male),
  loves(M,W),
  home(M,P2).

operation(defeat(M1,M2),2).
deleted(held_by(W,M2),defeat(M1,M2)).
precond(defeat(M1,M2),
  (/current_place(M2,P),
   /current_place(W,P),
   /held_by(W,M2),
   /current_place(M1,P))) :-
   gender(M1,male),
   gender(M2,male),
   not (M1 == M2),
   loves(M1,W),
   home(M2,P).
```

A.3. The behavioural schema

A.3.1. Goal-inference rules

```
% For the ravisher

sit_obj(M2,
  (current_place(W,P3),not current_place(M1,P3)),
  (current_place(W,P2))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  place(P3),
  not home(P3,_),
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).

% For the protector

sit_obj(M1,
  (current_place(W,P2), believes(M1,not loves(W,M2))),
  (current_place(W,P1), not held_by(W,M1))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  home(M1,P1),
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).

sit_obj(M1,
  (current_place(W,P2), believes(M1,loves(W,M2))),
  (current_place(W,P1), held_by(W,M1))) :-
  gender(M1,male),
  gender(M2,male),
  gender(W,female),
  not (M1 == M2),
  home(M1,P1),
  home(W,P1),
  home(M2,P2),
  not (P1 == P2).
```

A.3.2. Default beliefs (only for the protector)

```
belief(M1,loves(W,M2),S) :-
  gender(M1,male),
  gender(M2,male),
  not (M1 == M2),
  gender(W,female),
  home(M2,P2),
  once(current_place(W,P2),S),
  not (state(S2,S),holds(held_by(W,M2),S2)).

belief(M1,not loves(W,M2),S) :-
  gender(M1,male),
```

```
  gender(M2,male),
  not (M1 == M2),
  gender(W,female),
  home(M2,P2),
  once((current_place(W,P2),held_by(W,M2)),S).
```

A.3.3. Feelings at an indicated state

```
emotional_condition(C1,absolutely_happy,S) :-
  emotional_condition(C1,happy,S),
  not (state(Si,S), holds(held_by(C1,_),Si)).

emotional_condition(C1,happy,S) :-
  character(C1),
  loves(C1,C2),
  not (character(C3),
       not (C3==C2),
       holds(held_by(C1,C3),S)),
  once((holds(current_place(C1,P),S),
        holds(current_place(C2,P),S))).

emotional_condition(C,bored,S) :-
  character(C),
  not emotional_condition(C,happy,S).
```

A.4. Example initial state

A.4.1. Fixed properties

```
place(forest).
place(palace).
place('Lanka').
character('Sita').
gender('Sita',female).
home('Sita',palace).
character('Rama').
gender('Rama',male).
home('Rama',palace).
character('Ravana').
gender('Ravana',male).
home('Ravana','Lanka').
```

A.4.2. Varying properties

```
current_place('Sita',forest).
current_place('Rama',palace).
current_place('Ravana','Lanka').
loves('Rama','Sita').
loves('Ravana','Sita').
loves('Sita','Rama').
```