Representing RCC relations in temporal logic

Author:
Tim Harteveld

Supervisor:
Dr. ir. Jan Broersen

January 30, 2015
15 ECTS

Abstract:
In this article we attempt to represent a spatial logic with a temporal logic in such a way that the spatial relations and the temporal relations can be used like a spatio-temporal logic. The spatial logic we try to represent is the Region Connection Calculus (RCC). The temporal logics we will discuss are Linear-time Temporal Logic (LTL), Computation Tree Logic (CTL) and Alternating-time Temporal Logic (ATL). We also discuss the available model checkers for these temporal logics and give an example of the use of the spatio-temporal logic. We conclude that LTL can only represent some RCC relations. CTL can represent all RCC relations but temporal reasoning isn't possible anymore. ATL can represent all RCC relations and temporal reasoning is still possible.

Contents

1 Introduction

A lot of reasoning in artificial intelligence is based on logic. Logics about time are called temporal logics. They make it possible to reason about time. Logics about space are called spatial logics. They make it possible to reason about space. Combining these two logics into a spatio-temporal logic makes it possible to reason about space changing over time [6]. This makes it possible to reason about situations that involve both temporal and spatial aspects. In this article we want to represent spatial relations in a temporal logic. In this way the existing model checkers for the temporal logic can be used to reason about both time and space.

The spatial relation we want to represent in a temporal logic is the Region Connection Calculus (RCC). RCC makes it possible to reason about relations between spatial regions. It can be translated to propositional modal logic [5]. We want to translate it into a modal temporal logic. This gives us the following research questions:

> Can the RCC relation be represented into a modal temporal logic? Can this be done in such a way that the spatial relations and the temporal relations can be used like a spatio-temporal logic?

We are going discuss three different temporal logics to see which one is most suited for the translation of RCC. We will also discuss the available model checkers for these temporal logics. The temporal logics we will discuss are Linear-time Temporal Logic (LTL), Computation Tree Logic (CTL) and Alternating-time Temporal Logic (ATL).

In section 2 we define the temporal logics LTL, CTL and ATL. In section 3 we discuss some of the available model checkers for these temporal logics. In section 4 we define RCC and we try to represent RCC in LTL, CTL and ATL. In section 5 we give an example. In this example we show how the representation of RCC in a temporal logic can be used in a parking scenario. In section 6 we conclude this article and formulate an answer to our research questions.

2 Temporal logics

2.1 Linear-time Temporal Logic

LTL is introduced in *The temporal logic of programs* [8]. It is a logic where time is represented as a single path. A definition of LTL is given in *Principles of model checking* [4]. The syntax of LTL consists of atomic propositions Π, the standard boolean operators and temporal operators. It is defined with the following grammar, where $p \in \Pi$.

$$\varphi ::= \top \mid \bot \mid p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \Diamond\varphi \mid \Box\varphi \mid \bigcirc\varphi \mid \varphi_1 \mathcal{U} \varphi_2$$

The precedence of the unary operators is equally strong. The unary operators take precedence over the binary operators. The operators $\wedge$ and $\vee$ are equally strong and $\mathcal{U}$ is stronger.

The intuitive meaning of the boolean operators is the same as in proposition logic. $\Box\varphi$ is pronounced "globally φ" and means φ will always hold. $\Diamond\varphi$ is pronounced "finally φ" and means that φ eventually will hold. $\bigcirc\varphi$ is pronounced

"next φ" and means φ will hold in the next state. $\varphi_1 \mathcal{U} \varphi_2$ is pronounced "φ_1 until φ_2" and means φ_1 will hold until φ_2 holds.

Now we give the semantic definition of LTL. A path τ is an infinite line of connected states A_i with $i \geq 0$. A_0 is the starting point of the path. Every state is a set of atomic propositions that are true in the state. τ satisfies an LTL formula φ if $\tau \vDash \varphi$. Where the $\vDash$ relation has the following properties.

$$
\begin{aligned}
&\tau \vDash \top \\
&\tau \nvDash \bot \\
&\tau \vDash p && \text{iff } p \in A_0 \\
&\tau \vDash \neg\varphi && \text{iff } \tau \nvDash \varphi \\
&\tau \vDash \varphi \wedge \psi && \text{iff } \tau \vDash \varphi \text{ and } \tau \vDash \psi \\
&\tau \vDash \varphi \vee \psi && \text{iff } \tau \vDash \varphi \text{ or } \tau \vDash \psi \\
&\tau \vDash \Diamond\varphi && \text{iff } \exists j \geq 0.\ \tau[j \dots] \vDash \varphi \\
&\tau \vDash \Box\varphi && \text{iff } \forall j \geq 0.\ \tau[j \dots] \vDash \varphi \\
&\tau \vDash \bigcirc\varphi && \text{iff } \tau[1 \dots] \vDash \varphi \\
&\tau \vDash \varphi \mathcal{U} \psi && \text{iff } \exists j \geq 0.\ \tau[j \dots] \vDash \psi \text{ and } \tau[i \dots] \vDash \varphi, \text{ forall } 0 \leq i < j
\end{aligned}
$$

Where $\tau[i \dots]$ is the path τ without the first i states. The starting point is then A_i. Note that for $\bigcirc\varphi$ the formula φ must only hold in the state A_1, but to verify this the whole path $\tau[1 \dots]$ must be given. when φ contains temporal operators the whole path is needed to verify φ.

The semantics of LTL don't have to be in terms of paths, but can also be given in terms of transition systems. A transition system is a tuple $T = (Q, q_0, \delta, \Pi, l)$. Where Q is a set of states. $q_0 \in Q$ is the initial state. When $q \in Q$ and $0 < j \leq d(q)$ then $\delta(q, j) \in Q$ is a transition function. Where $d(q)$ is the number of possible transitions in q. There is a transition between q and $\delta(q, j)$ and in each state there is at least one transition possible. We identify the transitions with the number j. Π is the finite set of atomic propositions. l is the labelling function. For each state $q \in Q$, a set $l(q) \subseteq \Pi$ of propositions is true at q. An LTL formula must hold for every path in a transition system. A path in a transition system is an infinite sequence of states $q_0, q_1, \dots$ such that $\forall i \geq 0\ \delta(q_i, j) = q_{i+1}$ for some $0 < j \leq d(q_i)$. The LTL formula must hold for all possible paths of a given transition system.

2.2 Computation Tree Logic

CTL is a logic where time is represented as a tree. At every node there is a choice between different actions, which lead to different futures. Adding path quantifiers to LTL gives CTL*. CTL is a fragment of CLT* where a path operator must be followed with a temporal operator.

A definition of CTL is given in *Principles of model checking* [4]. With path quantifiers you can reason with different futures of paths. $\exists$ means there exists a path where φ holds and $\forall$ means that φ must hold on all paths. It is defined with a finite set of propositions Π in the following grammar, where $p \in \Pi$:

$$\varphi ::= \top \mid \bot \mid p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists\psi \mid \forall\psi$$
$$\psi ::= \Diamond\varphi \mid \Box\varphi \mid \bigcirc\varphi \mid \varphi_1 \mathcal{U} \varphi_2$$

The precedence fore CTL is the same as for LTL. The precedence of the unary operators is equally strong. The unary operators take precedence over the binary operators. The operators $\wedge$ and $\vee$ are equally strong and $\mathcal{U}$ is stronger.

The intuitive meaning of the boolean operators is the same as in proposition logic. The temporal operators always come in pairs. They have the following intuitive meanings. $\forall\Diamond\varphi$ is pronounced "φ is inevitable" and means that in every future φ holds at some point. $\exists\Diamond\varphi$ is pronounced "φ holds potentially" and means that there is a future where φ holds at some point. $\forall\Box\varphi$ is pronounced "invariantly φ" and means that in all futures φ always holds. $\exists\Box\varphi$ is pronounced "potentially always φ" and means that there is a future where φ always holds. $\forall\bigcirc\varphi$ is pronounced "for all paths next φ" and means that in all futures φ holds in the next stats. $\exists\bigcirc\varphi$ is pronounced "for some path next φ" and means that there is a future where φ holds in the next state. $\forall\varphi_1\mathcal{U}\varphi_2$ is pronounced "for all paths φ_1 until φ_2" and means that in all futures φ_1 holds until φ_2 holds. $\exists\varphi_1\mathcal{U}\varphi_2$ is pronounced "for some path φ_1 until φ_2" and means that there is a future where φ_1 holds until φ_2 holds.

Now we give the semantic definition of CTL. A tree π is an infinite set of states. Every state is a set of the atomic propositions that are true in this state. The first state of the tree is the root r. Every state is connected with other states which are called it's children. There are no cycles in the tree. π satisfies an LTL formula φ if $\pi \vDash \varphi$. The function $p(\pi)$ gives a set of all possible paths on tree π. A path is possible when $A_0 = r$ and A_{i+1} is a child of A_i. The $\vDash$ relation has the following properties.

$\pi \vDash \top$	
$\pi \nvDash \bot$	
$\pi \vDash p$	iff $p \in r$
$\pi \vDash \neg\varphi$	iff $\pi \nvDash \varphi$
$\pi \vDash \varphi \wedge \psi$	iff $\pi \vDash \varphi$ and $\pi \vDash \psi$
$\pi \vDash \varphi \vee \psi$	iff $\pi \vDash \varphi$ or $\pi \vDash \psi$
$\pi \vDash \exists\varphi$	iff $\tau \vDash \varphi$ for some $\tau \in P(\pi)$
$\pi \vDash \forall\varphi$	iff $\tau \vDash \varphi$ for all $\tau \in P(\pi)$

$\tau \vDash \Diamond\varphi$	iff $\exists j \geq 0. \tau[j] \vDash \varphi$
$\tau \vDash \Box\varphi$	iff $\forall j \geq 0. \tau[j] \vDash \varphi$
$\tau \vDash \bigcirc\varphi$	iff $\tau[1] \vDash \varphi$
$\tau \vDash \varphi\mathcal{U}\psi$	iff $\exists j \geq 0. (\tau[j] \vDash \psi$ and $(\forall 0 \leq k < j. \tau[k] \vDash \varphi))$

Here $\tau[j]$ gives a tree with the jst state in the path τ as the root of the tree.

The semantics of CTL don't have to be in terms of trees, but can also be given in terms of transition systems. A tree in transition system $T = (Q, q_0, \delta, \Pi, l)$ has it's root at q_0 and $\forall i \geq 0$ has children $\delta(q_i, j)$ for all j from 1 to $d(q_i)$.

2.3 Alternating-time Temporal Logic

ATL is a logic that makes it possible to reason about time with multiple agents. ATL uses a transition system similar to the one used for LTL and CTL called a game structure [3]. A game structure has multiple players, this makes it possible to reason with multiple agents. A game structure is a tuple $S = (k, Q, q_0, \delta, \Pi, l)$. Where $k \in \mathbb{N}$ is the number of players and $k \geq 1$. We write $\sum = \{1, \ldots, k\}$ for the set of players. Q is the set of states. $q_0 \in Q$ is the initial state. When $q \in Q$ and $\langle j_1, \ldots, j_k \rangle \in D(q)$ then $\delta(q, j_1, \ldots, j_k) \in Q$ is a transition function. Where $D(q)$ is a set of possible move vectors. A move vector at q is a tuple $\langle j_1, \ldots, j_k \rangle$ such that $1 \leq j_a \leq d_a(q)$. $d_a(q)$ is the number of possible transitions in q for player $a \in \sum$. There is a transition between q and $\delta(q, j_1, \ldots, j_k)$ if the players can choose moves so that $\langle j_1, \ldots, j_k \rangle \in D(q)$. In each state there is at least one transition possible. Π is the finite set of atomic propositions. l is the labelling function. For each state $q \in Q$, a set $l(q) \subseteq \Pi$ of propositions is true at q.

A definition of ATL is given in the article *Alternating-time temporal logic* [3]. For every temporal operator a set of players is given. It is defined in the following grammar, where $p \in \Pi$ and $A \subseteq \sum$.

$$\varphi ::= \top \mid \bot \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \langle\langle A \rangle\rangle \Diamond \varphi \mid \langle\langle A \rangle\rangle \Box \varphi \mid \langle\langle A \rangle\rangle \bigcirc \varphi \mid \langle\langle A \rangle\rangle \varphi_1 \mathcal{U} \varphi_2$$

$\langle\langle \{a_1, \ldots, a_l\} \rangle\rangle$ can be written as $\langle\langle a_1, \ldots, a_l \rangle\rangle$ and $\langle\langle \varnothing \rangle\rangle$ can be written as $\langle\langle \rangle\rangle$. $[\![A]\!]\varphi$ is defined as $\neg\langle\langle A \rangle\rangle\neg\varphi$.

The intuitive meaning of the boolean operators is the same as in proposition logic. The intuitive meaning of $\langle\langle A \rangle\rangle\psi$ is that the set of players A can cooperate to make ψ true. The intuitive meaning of $[\![A]\!]\psi$ is that the set of players A cannot avoid to make ψ true. The temporal operators have the following intuitive meanings. $\langle\langle A \rangle\rangle \Box \varphi$ means players A can cooperate to make φ always hold. $\langle\langle A \rangle\rangle \Diamond \varphi$ means players A can cooperate to make φ eventually hold. $\langle\langle A \rangle\rangle \bigcirc \varphi$ means players A can cooperate to make φ hold in the next state. $\langle\langle A \rangle\rangle \varphi_1 \mathcal{U} \varphi_2$ means players A can cooperate to make φ_1 hold until φ_2 holds.

To give the semantic definition of ATL we first define the notion of strategies. A strategy for player $a \in \sum$ is the function f_a. Given a nonempty finite state sequence $\lambda \in Q^+$ the function f_a gives a natural number such that $f_a(\lambda) \leq d_a(q)$ where q is the last state of λ. We write $F_A = \{f_a | a \in A\}$ for the set of strategies of the players in A. We define $out(q, F_A)$ to be the state sequence followed when q is the initial state and all players in A follow their strategies in F_A. thus λ is $out(q, F_A)$ if $q = q_0$ and for all positions $i \geq 0$ there is a move vector $\langle j_1, \ldots, j_k \rangle \in D(q_i)$ such that $j_a = f_a(\lambda[0, i])$ for all players $a \in A$ and $\delta(q_i, j_1, \ldots, j_k) = q_{i+1}$.

Now we can give the definition of the semantics of ATL. A state q satisfies an ATL formula φ if $q \vDash \varphi$. The $\vDash$ relation has the following properties.

$q \models \top$

$q \not\models \bot$

$q \models p$ iff $p \in l(q)$

$q \models \neg\varphi$ iff $q \not\models \varphi$

$q \models \varphi \wedge \psi$ iff $q \models \varphi$ and $q \models \psi$

$q \models \varphi \vee \psi$ iff $q \models \varphi$ or $q \models \psi$

$q \models \langle\!\langle A \rangle\!\rangle \Diamond \varphi$ iff there exists a set F_A such that for all $\lambda \in out(q, F_A)$, there is a $i \geq 0$ such that $\lambda[i] \models \varphi$.

$q \models \langle\!\langle A \rangle\!\rangle \Box \varphi$ iff there exists a set F_A such that for all $\lambda \in out(q, F_A)$ and all positions $i \geq 0$, we have $\lambda[i] \models \varphi$.

$q \models \langle\!\langle A \rangle\!\rangle \bigcirc \varphi$ iff there exists a set F_A such that for all $\lambda \in out(q, F_A)$ we have $\lambda[1] \models \varphi$.

$q \models \langle\!\langle A \rangle\!\rangle \varphi \,\mathcal{U}\, \psi$ iff there exists a set F_A such that for all $\lambda \in out(q, F_A)$, there exists a position $i \geq 0$ such that $\lambda[i] \models \varphi_2$ and for all positions $0 \leq j \leq i$, we have $\lambda[j] \models \varphi_1$.

3 Model checking

A model checker is a program that given a model can answer questions about that model. For LTL and CTL a model checker can check whether an LTL or CTL formula is true or false given a certain transition system. A model checker for ATL can check whether an ATL formula is true or false given an game structure.

Model checking for LTL and CTL is often combined into a single model checker. There are many model checkers for LTL and CTL, but there are not many model checkers for ATL. The only model checker for ATL we are aware of is called Mocha. In section 3.1 we give a description of a model checker for LTL and CTL. In section 3.2 we give a description the model checker Mocha for ATL.

3.1 MCheck

MCheck [10] is a simple model checker for LTL and CTL. The program is a .JAR file. This makes it easy to implement in java programs and nothing needs to be installed.

MCheck has no GUI interface but is still simple to use. The command `mch test.txt` will verify the LTL and CTL formulas given in `test.txt` for the model given in `test.txt`. A model is given by writing for each state the name followed by the states to which the state has transitions and the propositions that are true in the state. The initial state is indicated with '>' before the name of the state. Figure 1 gives an example of how models are represented in MCheck. The LTL and CTL formulas are written as expected, where the temporal operators are given with capital letters.

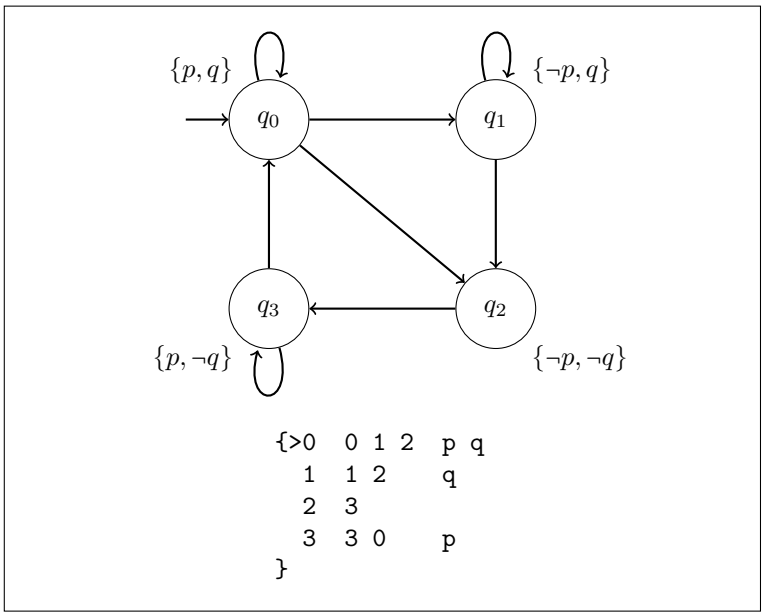


Figure 1: How a model is represented in MCheck

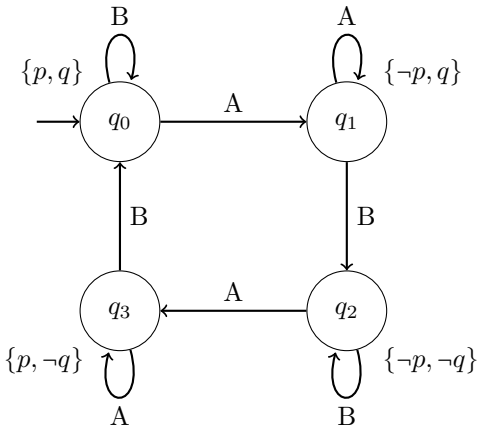
3.2 Mocha

There are two versions of Mocha, cMocha [2] and jMocha [1]. cMocha is the first version and is mostly written in C. jMocha is the second version and is mostly written in Java. On Windows jMocha didn't fully work, on Linux it did, cMocha didn't work at all[1].

jMocha can only do invariant and refinement checking. Invariant checking can only check formulas of the form $\langle\langle A \rangle\rangle \Box \varphi$ where φ doesn't contain any other temporal operators. Nesting temporal operators is not possible. Refinement checking is checking whether the traces generated by one model is a subset of the traces generated by another model.

Giving a model to Mocha is not as simple as it is in MCheck. A set of variables is given instead of a set of states. Each possible combination of values of the variables is a state. The transitions between the states are given by allowing each player to only change certain variables. Each variable can only be controlled by one player. The strategy of the player is given by formulas that assign values to the variables. These formulas can use variables controlled by other players. Every game structure can be represented in this way, but the translation isn't always straightforward. Figure 2 gives an example of how models are represented in Mocha.

[1]Installing cMocha on Linux gave the following problems. tcl version 8.6 doesn't work with Tix version 8.4.3. The code in `until.h` has `(obj)=0` instead of `(obj)==0` (line 175 and 185). File `invMain.c` gives and error (line 127). The file `configure` of mocha-1.0.1 has an apostrophe to much (line 850). File `prsLex.c` gives an error (line 1527).




```
module A is
  interface p : bool
  external q : bool

  atom contp controls p reads q
    init
      [] true -> p' := true
    update
      [] q -> p' := false
      [] ~q -> p' := true

module B is
  interface q : bool
  external p : bool

  atom contq controls q reads p
    init
      [] true -> q' := true
    update
      [] p -> q' := false
      [] ~p -> q' := true
```

Figure 2: How a model is represented in Mocha

4 Spatial reasoning

Now that we have defined LTL, CTL and ATL and described model checkers for these temporal logics we want to represent spatial information with these temporal logics. Every temporal step between states can be seen as a step in spatial space. What can be expressed with the temporal logics is however not very useful for spatial reasoning. That is why we represent RCC with the temporal logics. First we will define RCC and then we try to represent it with the temporal logics.

4.1 Region Connection Calculus

RCC is a way to represent relations between regions. The definition for RCC is based on the $C(X,Y)$ relation which means X connects with Y [9]. Where X and Y are spatial regions. This relation is reflexive and symmetric, thus X is connected with itself and if X is connected with Y then is Y connected with X. All other relations are defined with $C(x,y)$ in the following way:

$$
\begin{aligned}
\mathsf{DC}(X,Y) &:= \neg C(X,Y) \\
\mathsf{P}(X,Y) &:= \forall_Z[C(Z,X) \rightarrow C(Z,Y)] \\
\mathsf{PP}(X,Y) &:= \mathsf{P}(X,Y) \wedge \neg \mathsf{P}(Y,X) \\
\mathsf{EQ}(X,Y) &:= \mathsf{P}(X,Y) \wedge \mathsf{P}(Y,X) \\
\mathsf{O}(X,Y) &:= \exists_Z[\mathsf{P}(Z,X) \wedge \mathsf{P}(Z,Y)] \\
\mathsf{PO}(X,Y) &:= \mathsf{O}(X,Y) \wedge \neg \mathsf{P}(X,Y) \wedge \neg \mathsf{P}(Y,X) \\
\mathsf{DR}(X,Y) &:= \neg \mathsf{O}(X,Y) \\
\mathsf{TPP}(X,Y) &:= \mathsf{PP}(X,Y) \wedge \exists_Z[\mathsf{EC}(Z,X) \wedge \mathsf{EC}(Z,Y)] \\
\mathsf{EC}(X,Y) &:= C(x,y) \wedge \neg \mathsf{O}(X,Y) \\
\mathsf{NTPP}(X,Y) &:= \mathsf{PP}(X,Y) \wedge \neg \exists_Z[\mathsf{EC}(Z,X) \wedge \mathsf{EC}(Z,Y)] \\
\mathsf{P}^{-1}(X,Y) &:= \mathsf{P}(Y,X) \\
\mathsf{PP}^{-1}(X,Y) &:= \mathsf{PP}(Y,X) \\
\mathsf{TPP}^{-1}(X,Y) &:= \mathsf{TPP}(Y,X) \\
\mathsf{NTPP}^{-1}(X,Y) &:= \mathsf{NTPP}(Y,X)
\end{aligned}
$$

$\mathsf{DC}(X,Y)$ stands for X is disconnected from Y, $\mathsf{P}(X,Y)$ stands for X is part of Y, $\mathsf{PP}(X,Y)$ stands for X is a proper part of Y, $\mathsf{EQ}(X,Y)$[2] stands for X is equivalent to Y, $\mathsf{O}(X,Y)$ stands for X overlaps Y, $\mathsf{PO}(X,Y)$ stands for X partially overlaps Y, $\mathsf{DR}(X,Y)$ stands for X is discrete from Y, $\mathsf{TPP}(X,Y)$ stands for X is a tangential proper part of Y, $\mathsf{EC}(X,Y)$ stands for X is externally connected with Y, $\mathsf{NTPP}(X,Y)$ stands for X is a non-tangential proper part of Y. For the inverse of ϕ the notation ϕ^{-1} is used, where $\phi \in \{\mathsf{P}, \mathsf{PP}, \mathsf{TPP}, \mathsf{NTPP}\}$.

In this article we will use RCC8. RCC8 only uses the relations DC, EC, PO, EQ, TPP, NTTP, TTP^{-1} and NTTP^{-1}. A visual representation of these relations is shown in Figure 3.

[2] In [9] $X = Y$ is used instead of $\mathsf{EQ}(X,Y)$.

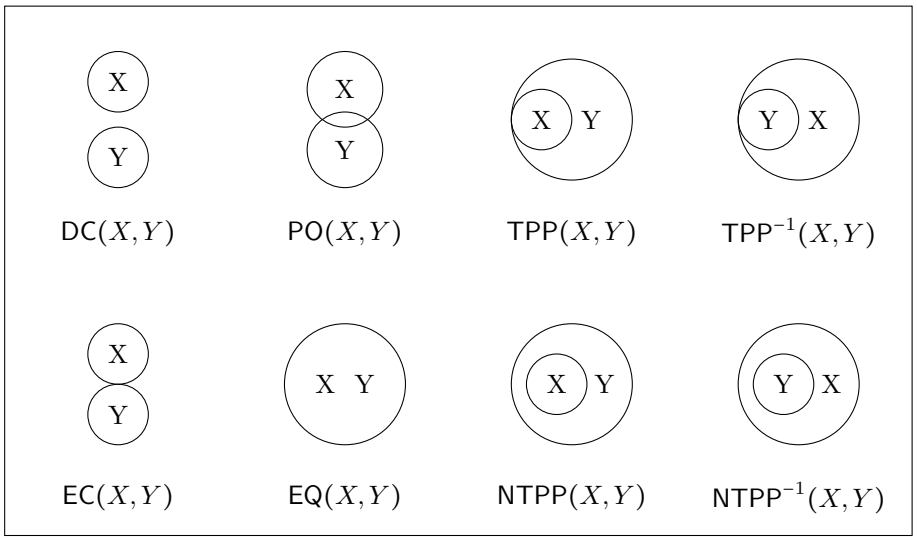


Figure 3: A visual representation of the RCC8 relations

There is a distinction between the interior and the boundary of a region. The interior of φ is denoted $i\varphi$. Looking at Figure 3 it is easy to see that the RCC8 relations can also be defined with the following set formulas:

$$\begin{aligned}
\mathsf{DC}(X,Y) &:= X \cap Y = \varnothing \\
\mathsf{EC}(X,Y) &:= X \cap Y \neq \varnothing \wedge iX \cap iY = \varnothing \\
\mathsf{PO}(X,Y) &:= X \nsubseteq Y \wedge Y \nsubseteq X \wedge iX \cup iY \neq \varnothing \\
\mathsf{EQ}(X,Y) &:= X = Y \\
\mathsf{TPP}(X,Y) &:= X \subset Y \wedge X \notin iY \\
\mathsf{NTPP}(X,Y) &:= X \subset iY
\end{aligned}$$

Modal Logics for Qualitative Spatial Reasoning [5] gives a translation from RCC constraints to multimodal propositional logic. We want a translation from RCC constraints to LTL, CTL and ATL. *On the Translation of Qualitative Spatial Reasoning Problems into Modal Logics* [7] gives a proof for the translation to multimodal propositional logic. I'm using a similar proof for the translation to the temporal logics.

We give a formal language called set expression. It is easier to use set expressions for our purposes then the set formulas given above. Set expressions s and t are with the following grammar. X, Y and Z denote the countable infinite set of variables.

$$s, t ::= X \mid \top \mid \bot \mid s \sqcup t \mid s \sqcap t \mid \overline{s} \mid \mathbf{I}s$$

Elementary set constraints have the form $s \doteq t$ or $s \neq t$. More complex set constraints can be constructed using the propositional connectives conjunction, disjunction and negation. If S and T are set constraints then so are $S \wedge T$, $S \vee T$ and $\neg S$.

The function d maps every set expression to a subset of a topological space. A topological space consist of a universe U and a set i of subset of U. d maps every set expression to a subset of U in the following way:

$$\begin{aligned} d(\bot) &= \varnothing & d(\top) &= U \\ d(s \sqcap t) &= d(s) \cap d(t) & d(s \sqcup t) &= d(s) \cup d(t) \\ d(\overline{s}) &= U \setminus d(s) & d(\mathbf{I}s) &= i(d(s)) \end{aligned}$$

A topological interpretation $\mathcal{I}$ is a triple $\mathcal{I} = (U, i, d)$. We write $\mathcal{I} \vDash C$ if topological interpretation $\mathcal{I}$ satisfies a set constraint C. The topological interpretation for elementary constraints is defined as below. The conjunction, disjunction and negation in complex topological interpretations are defined in their usual way.

$$\begin{aligned} \mathcal{I} &\vDash s \doteq t \text{ iff } d(s) = d(s) \\ \mathcal{I} &\vDash s \not\doteq t \text{ iff } d(s) \neq d(s) \end{aligned}$$

Now we can rewrite the RCC relations from set formulas into the set expressions:

$$\begin{aligned} \mathsf{DC}(X,Y) &:= X \sqcap Y \doteq \bot \\ \mathsf{EC}(X,Y) &:= X \sqcap Y \not\doteq \bot \wedge \mathbf{I}X \sqcap \mathbf{I}Y \doteq \bot \\ \mathsf{PO}(X,Y) &:= \mathbf{I}X \sqcap \mathbf{I}Y \not\doteq \bot \wedge X \sqcap \overline{Y} \not\doteq \bot \wedge \overline{X} \sqcap Y \not\doteq \bot \\ \mathsf{EQ}(X,Y) &:= X \doteq Y \\ \mathsf{TPP}(X,Y) &:= X \sqcap \overline{Y} \doteq \bot \wedge X \sqcap \overline{\mathbf{I}\overline{Y}} \not\doteq \bot \\ \mathsf{NTPP}(X,Y) &:= X \sqcap \overline{\mathbf{I}\overline{Y}} \doteq \bot \end{aligned}$$

The temporal logics LTL, CTL and ATL are not continues, they are based on transitions systems. This means that the spatial regions used in RCC must also be represented in a transition system $T_s = (Q, q_0, \delta, \Pi, l)$ that we call spatial transition system. The universe U will be the set of all states Q. q_0 is a state that's not part of any region. δ is the reachability between states. typically, if two states are next to each other in the spatial space then they are connected in both ways. A region will be a set of states where a certain proposition is true. The distinction between the interior and the boundary of a region is made with the reachability of states. The interior has no transitions to states outside the interior. This makes it possible to say that something holds inside the interior of are region with the expression it holds in all reachable states, starting inside the region. Figure 4 gives an example of a spatial transition system. Dotted lines show which part of the transition system corresponds to the regions. The set of true propositions are given for each state. The regions are represented in a five by nine grid of states. Typically a lot more states are used, but for the clarity of this example we used the minimal number of states necessary. It is also not required for the states to be in a grid.

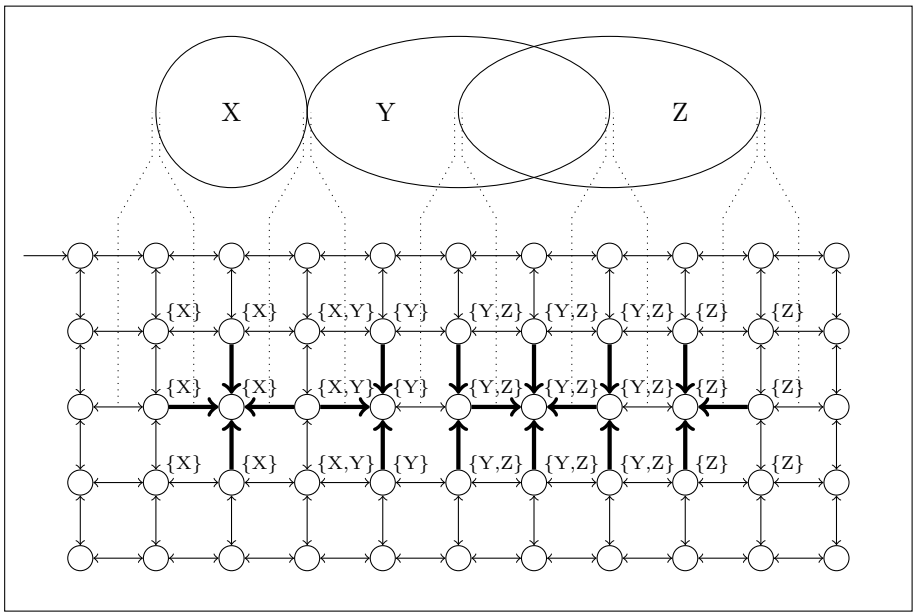


Figure 4: Example of three regions represented in a spatial transition system

4.2 Representing RCC in LTL

Now that we have set expressions for the RCC8 relations we can try to give a translation from these set expressions to LTL formulas. The function α makes this translation by mapping every set expression to an LTL formula in the following way:

$$\begin{aligned} \alpha(\bot) &= \bot & \alpha(\top) &= \top \\ \alpha(s \sqcap t) &= \alpha(s) \wedge \sigma(t) & \alpha(s \sqcup t) &= \alpha(s) \vee \alpha(t) \\ \alpha(\overline{s}) &= \neg\alpha(s) & \alpha(\mathbf{I}s) &= \Box\alpha(s) \end{aligned}$$

Most of these mappings are very straightforward. The mapping of the S4 operator $\alpha(\mathbf{I}s)$ to $\Box\alpha(s)$ isn't directly obvious. $\mathbf{I}$ is a S4 operator, equivalent to the $\Box$ operator in a S4 frame. LTL is a reflexive and transitive and thus has a S4 frame. Because LTL is S4, $\alpha(\mathbf{I}s)$ can be mapped to $\Box\alpha(s)$.

We now have a translation from set expressions to LTL formulas. However the translation from set constraints to LTL formulas with the functions α gives a problem. The translation from $s \doteq \top$ to s must hold everywhere in the universe is simple to do with the $\Box$ operator. This translation is given below. But translating $s \not\doteq \top$ to there is a point in the universe where s doesn't hold is not possible. The $\Diamond$ operator seems like a good option, but $\Diamond\varphi$ means on every path φ eventually holds and not there is a path where φ eventually holds.

$$\alpha(s \doteq \top) = \Box\alpha(s) \qquad \alpha(s \doteq \bot) = \Box\neg\alpha(s)$$

We can only translate set constraints where $\not\doteq$ is not used. Only the RCC set expressions DC, EQ and NTPP don't use $\not\doteq$. Using the set expressions of these RCC relations gives the following result:

$$\begin{aligned}
\alpha(\mathsf{DC}(X,Y)) &= \Box\neg(X \wedge Y) \\
\alpha(\mathsf{EQ}(X,Y)) &= \Box(\neg X \vee Y) \wedge \\
&\quad\ \Box(X \vee \neg Y) \\
\alpha(\mathsf{NTPP}(X,Y)) &= \Box(\neg X \vee \Box Y)
\end{aligned}$$

LTL can only represent the RCC relations DC, EQ and NTPP and not all RCC8 relations. This is because in LTL the formulas must hold for all paths. A formula that holds for some paths cannot be represented in LTL but can be represented in CTL.

4.3 Representing RCC in CTL

We will show that in CTL you can represent RCC. The proof is similar to the one given for LTL, but now we can represent formulas that hold for some paths. The function β maps every set expression formula to a CTL formula in the following way:

$$\begin{aligned}
\beta(\bot) &= \bot & \qquad \beta(\top) &= \top \\
\beta(s \sqcap t) &= \beta(s) \wedge \beta(t) & \qquad \beta(s \sqcup t) &= \beta(s) \vee \beta(t) \\
\beta(\overline{s}) &= \neg\beta(s) & \qquad \beta(\mathbf{I}s) &= \forall\Box\beta(s)
\end{aligned}$$

Just like with LTL the S4 operator $\mathbf{I}s$ is translated with the $\Box$ operator. It is translated to $\forall\Box\beta(s)$ because s must hold everywhere in the region. Because there are no transitions from the interior of a region to the outside s must always hold on all paths.

In CTL we can represent set constraints in the following way:

$$\begin{aligned}
\beta(s \doteq \top) &= \forall\Box\beta(s) & \qquad \beta(s \doteq \bot) &= \forall\Box\neg\beta(s) \\
\beta(s \not\doteq \top) &= \exists\Diamond\neg\beta(s) & \qquad \beta(s \not\doteq \bot) &= \exists\Diamond\beta(s)
\end{aligned}$$

Using β on the RCC set expressions gives the following result:

$$
\begin{aligned}
\beta(\mathsf{DC}(X,Y)) &= \forall\Box\neg(X \wedge Y) \\
\beta(\mathsf{EC}(X,Y)) &= \forall\Box\neg(\forall\Box X \wedge \forall\Box Y) \wedge \\
&\quad \exists\Diamond(X \wedge Y) \\
\beta(\mathsf{PO}(X,Y)) &= \exists\Diamond(\forall\Box X \wedge \forall\Box Y) \wedge \\
&\quad \exists\Diamond(X \wedge \neg Y) \wedge \\
&\quad \exists\Diamond(\neg X \wedge Y) \\
\beta(\mathsf{EQ}(X,Y)) &= \forall\Box(\neg X \vee Y) \wedge \\
&\quad \forall\Box(X \vee \neg Y) \\
\beta(\mathsf{TPP}(X,Y)) &= \forall\Box(\neg X \vee Y) \wedge \\
&\quad \exists\Diamond(X \wedge \neg\forall\Box Y) \\
\beta(\mathsf{NTPP}(X,Y)) &= \forall\Box(\neg X \vee \forall\Box Y)
\end{aligned}
$$

We now have a representation of RCC8 in CTL. However the representation of regions in the transition system makes temporal reasoning not possible. There is no distinction between temporal transitions and spatial transitions. We need to make a distinction between temporal and spatial operators.

4.4 Representing RCC in ATL

Just like in CTL, RCC can be represented in ATL. But we can make a distinction between temporal and spatial operators by making a distinction between players for temporal reasoning and players for spatial reasoning. γ maps every set expression formula to an ATL formula in the following way:

$$
\begin{aligned}
\gamma(\bot) &= \bot & \qquad \gamma(\top) &= \top \\
\gamma(s \sqcap t) &= \gamma(s) \wedge \gamma(t) & \qquad \gamma(s \sqcup t) &= \gamma(s) \vee \gamma(t) \\
\gamma(\overline{s}) &= \neg\gamma(s) & \qquad \gamma(\mathbf{I}s) &= \langle\!\langle A_s \rangle\!\rangle \Box\gamma(s)
\end{aligned}
$$

A_s is the set of players active in s.
Set constraints are represented in the following way:

$$
\begin{aligned}
\gamma(s \doteq \top) &= \langle\!\langle A_s \rangle\!\rangle \Box\gamma(s) & \qquad \gamma(s \doteq \bot) &= \langle\!\langle A_s \rangle\!\rangle \Box\neg\gamma(s) \\
\gamma(s \not\doteq \top) &= \langle\!\langle A_s \rangle\!\rangle \Diamond\neg\gamma(s) & \qquad \gamma(s \not\doteq \bot) &= \langle\!\langle A_s \rangle\!\rangle \Diamond\gamma(s)
\end{aligned}
$$

Using γ on the RCC set expressions gives the following result, where region X is defined by player x and region Y by player y.

$$
\begin{aligned}
\gamma(\mathsf{DC}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Box \neg (X \wedge Y) \\
\gamma(\mathsf{EC}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Box \neg (\langle\!\langle x\rangle\!\rangle \Box X \wedge \langle\!\langle y\rangle\!\rangle \Box Y) \wedge \\
&\quad \langle\!\langle x,y\rangle\!\rangle \Diamond (X \wedge Y) \\
\gamma(\mathsf{PO}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Diamond (\langle\!\langle x\rangle\!\rangle \Box X \wedge \langle\!\langle y\rangle\!\rangle \Box Y) \wedge \\
&\quad \langle\!\langle x,y\rangle\!\rangle \Diamond (X \wedge \neg Y) \wedge \\
&\quad \langle\!\langle x,y\rangle\!\rangle \Diamond (\neg X \wedge Y) \\
\gamma(\mathsf{EQ}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Box (\neg X \vee Y) \wedge \\
&\quad \langle\!\langle x,y\rangle\!\rangle \Box (X \vee \neg Y) \\
\gamma(\mathsf{TPP}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Box (\neg X \vee Y) \wedge \\
&\quad \langle\!\langle x,y\rangle\!\rangle \Diamond (X \wedge \neg \langle\!\langle y\rangle\!\rangle \Box Y) \\
\gamma(\mathsf{NTPP}(X,Y)) &= \langle\!\langle x,y\rangle\!\rangle \Box (\neg X \vee \langle\!\langle y\rangle\!\rangle \Box Y)
\end{aligned}
$$

Making a distinction between temporal players and spatial players makes it possible to represent both temporal and spatial information in a transition system. Figure 5 gives an example of how this is done. Every node in the temporal transition system is replaced with the set of nodes of the spatial transition system. These nodes are connected for the spatial player. In the example these are q_0, q_1 and q_2. All these nodes are connected for the temporal player with the node on the same location in the next state. In the example a few of these connections are represented by dotted lines.

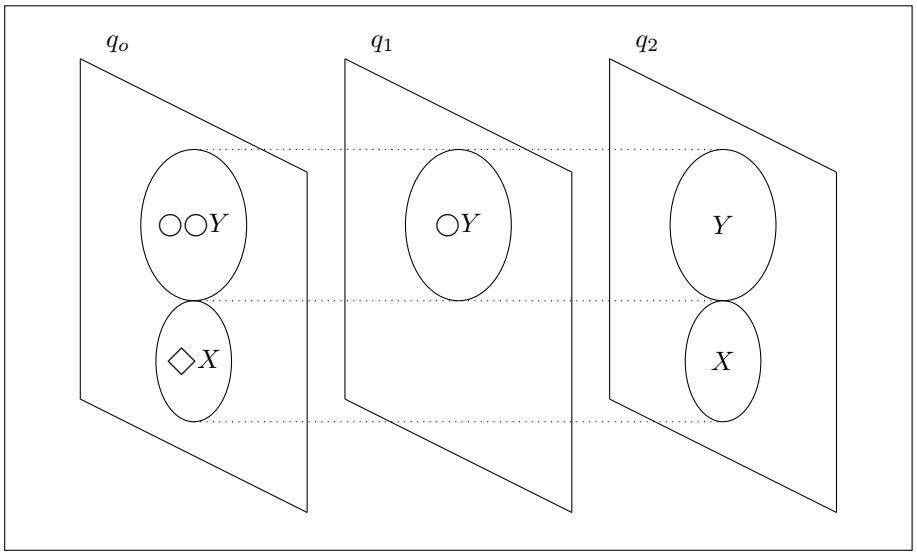


Figure 5: The representation of three temporal states in ATL

5 Parking example

In this example we show how the combination of ATL and RCC can be used to reason about parking fees. There are often different parking zones in a city with

different rates. These rates can also change over time. We will give a fictional city with parking zones and changing rates. Within this example we will try to answer some questions like in what regions you can park for what rates.

Figure 6 shows a map of the fictional city and names the different regions in the city. There are different parking zones with different parking fees in the city. There is a rate A, B and C. We denote free parking with 0. The rates differ for weekday, Saturdays and Sunday's. Figure 7 shows the transition system for this situation and the rates for the different zones and the different days.

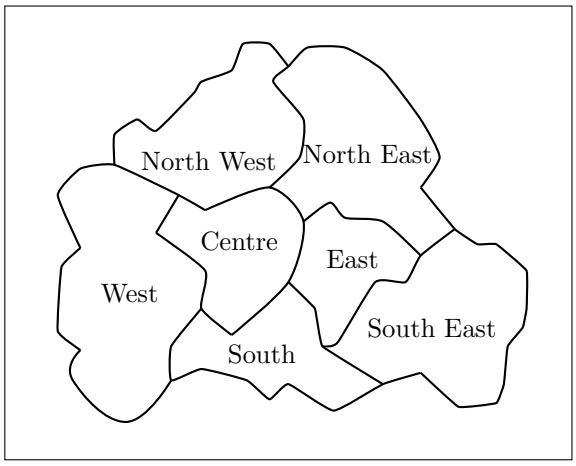


Figure 6: The city layout

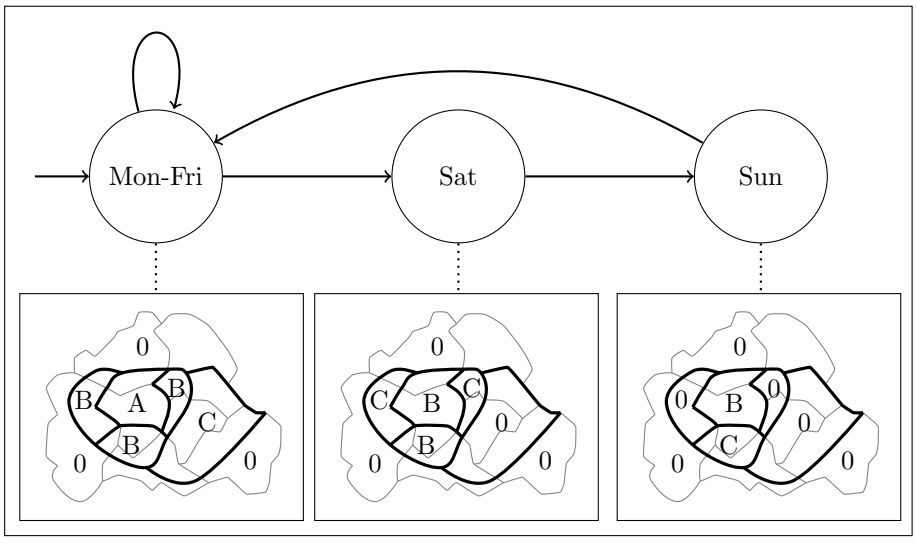


Figure 7: The parking zones and there rates

We now want to represent some questions about the parking in this city in ATL formula's. To do this we use the following players. The temporal player is called *days*, the spatial player for the city regions *city* and the spatial player for

the parking zones *zone*. When the questions are translated to ATL formula's we can use a model checker to get the answer.

Is there a moment when I can park in the Centre for free?

It is possible to park in the Centre for free if the Centre region overlaps with a zone with rate 0. "There is a moment" can be translated with the temporal $\Diamond$ operator. Note that if we want to know whether we can park in a region for a certain rate we only have to check whether the region and the zone partially overlap. We don't have to check whether the zone and the region are equivalent or are some proper part of each other, because in this example there are no zones and regions that are equivalent or are a proper part of each other. The formula for this question becomes $\Diamond\mathsf{PO}(Centre, 0)$, or in ATL:

$$\begin{aligned}\langle\!\langle days\rangle\!\rangle\Diamond(&\langle\!\langle city, zone\rangle\!\rangle\Diamond(\langle\!\langle city\rangle\!\rangle\Box Centre \wedge \langle\!\langle zone\rangle\!\rangle\Box 0) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(Centre \wedge \neg 0) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(\neg Centre \wedge 0))\end{aligned}$$

It is possible to park for free in the Centre at any moment. On every day there is a parking fee for all the zones that overlap the Centre region.

Can you park in South with rate B for two days?

Just like with the previous question, if we want to know whether we can park in a region for a certain rate we only have to check whether the region and the zone partially overlap. But now we want to check it for the current state and the next. We can check the next state using the temporal $\bigcirc$ operator. The formula for this question becomes $\mathsf{PO}(South, B) \wedge \bigcirc\mathsf{PO}(South, B)$, or in ATL:

$$\begin{aligned}(&\langle\!\langle city, zone\rangle\!\rangle\Diamond(\langle\!\langle city\rangle\!\rangle\Box South \wedge \langle\!\langle zone\rangle\!\rangle\Box B) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(South \wedge \neg B) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(\neg South \wedge B)) \wedge\\ (&\langle\!\langle days\rangle\!\rangle\bigcirc\langle\!\langle city, zone\rangle\!\rangle\Diamond(\langle\!\langle city\rangle\!\rangle\Box South \wedge \langle\!\langle zone\rangle\!\rangle\Box B) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(South \wedge \neg B) \wedge\\ &\langle\!\langle city, zone\rangle\!\rangle\Diamond(\neg South \wedge B)))\end{aligned}$$

We start on a weekday, as shown in Figure 7. It is possible to park in South for rate B on every weekday and on Saturday. You can park in South for two days with rate B.

Is there a moment when you can park for free next to the Centre?

If you can park for free next to the Centre then there must be a free parking zone just outside of the Centre region. This is the case when the region and the zone are externally connected, but also when they partially overlap. The temporal

$\diamondsuit$ operator is again used for translating "there is a moment". The formula for this question becomes $\diamondsuit(\mathsf{EC}(Centre, 0) \vee \mathsf{PO}(Centre, 0))$, or in ATL:

$$\begin{aligned}\langle\!\langle days \rangle\!\rangle \diamondsuit ((&\langle\!\langle city, zone \rangle\!\rangle \diamondsuit (\langle\!\langle city \rangle\!\rangle \Box Centre \wedge \langle\!\langle zone \rangle\!\rangle \Box 0) \wedge \\ &\langle\!\langle city, zone \rangle\!\rangle \diamondsuit (Centre \wedge \neg 0) \wedge \\ &\langle\!\langle city, zone \rangle\!\rangle \diamondsuit (\neg Centre \wedge 0)) \vee \\ &(\langle\!\langle city, zone \rangle\!\rangle \Box \neg (\langle\!\langle city \rangle\!\rangle \Box Center \wedge \langle\!\langle zone \rangle\!\rangle \Box 0) \wedge \\ &\langle\!\langle city, zone \rangle\!\rangle \diamondsuit (Centre \wedge 0)))\end{aligned}$$

We already know that you can never park in the Centre for free. Meaning there is no partial overlap between the centre and a free parking zone. But there is a moment when the Centre and a free parking zone are externally connected. Namely on Sunday you can park for free just east and just west of the Centre. There is a moment when you can park for free next to the Centre.

6 Conclusion

In this article we asked whether RCC relations could be represented into a temporal logic and whether this could be done in such a way that the spatial regions and the temporal relations can be used like in a spatio-temporal logic. We have found that LTL can only represent the RCC relations DC, EQ and NTPP. We found that CTL can represent all the RCC relations, but it is not possible to also use temporal relations. We found that in ATL both the spatial RCC relations and temporal relations can be used like in a spatio-temporal logic. This is possible by making a distinction between temporal and spatial players.

We wanted to represent a spatial logic into a temporal logic so that the existing model checker for the temporal logic could be used. However there is only one model checker for ATL we are aware of and it's difficult to use and install. There are a lot more model checkers for LTL and CTL available. A model checker for CTL could be used to check RCC models.

Because the model checker for ATL is difficult to use it might be better to develop a spatio-temporal model checker instead of using an existing temporal model checker. In this article we only discussed RCC as the spatial logic. Future research could focus on representing other spatial logics into temporal logics. Future research could also focus on other temporal logics to represent the spatial logics.

In artificial intelligence temporal logics is used to reason about time and spatial logics to reason about space. In this article we combined some of these temporal and spatial logics into a spatio-temporal logic. Making it possible to reason about space changing over time, like in the given parking example.

References

[1] R Alur, H Anand, F Ivancić, M Kang, M McDougall, BY Wang, L de Alfaro, TA Henzinger, B Horowitz, R Majumdar, FYC Mang, C Meyer-Krisch, M Minea, S Qadeer, SK Rajamani, and JF Raskin. *MOCHA user manual. Jmocha version 2.0.*

[2] R Alur, L De Alfaro, Th A Henzinger, SC Krishnan, FYC Mang, S Qadeer, SK Rajamani, and S Tasiran. *MOCHA user manual.*

[3] Rajeev Alur, Thomas A Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM (JACM)*, 49(5):672–713, 2002.

[4] Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*, volume 26202649. MIT press Cambridge, 2008.

[5] Brandon Bennett. Modal logics for qualitative spatial reasoning. *Logic Journal of IGPL*, 4(1):23–45, 1996.

[6] Roman Kontchakov, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. Spatial logic+ temporal logic=? In *Handbook of spatial logics*, pages 497–564. Springer, 2007.

[7] Werner Nutt. On the translation of qualitative spatial reasoning problems into modal logics. In *KI-99: Advances in Artificial Intelligence*, pages 113–124. Springer, 1999.

[8] Amir Pnueli. The temporal logic of programs. In *Foundations of Computer Science, 1977., 18th Annual Symposium on*, pages 46–57. IEEE, 1977.

[9] David A Randell, Zhan Cui, and Anthony G Cohn. A spatial logic based on regions and connection. *KR*, 92:165–176, 1992.

[10] Jeff Sember. Mcheck: A model checker for ltl and ctl formulas. 2005.