2 Definitions

This chapter presents the theoretical background and terminology used in this thesis. Section 2.1 presents the logical definitions used in this work followed by the probabilistic and statistical definitions in Section 2.2. These notions are required to the fully comprehension of the logics in here proposed as well as the PDL and Petri Net systems adopted in sections 2.3.1 and 2.3.2.

2.1 Basic logical definitions

The definitions in this section follows the presentation in the work of Goldblatt (1992*a*). We will use the notation of Backus-Naus form (BNF) to define the syntax of the languages. A BNF statement is in the form "$\langle$variable1$\rangle$ ::= expression" where "atom" is an element of the alphabet of the language, "::=" means "produces" or "is replaced by" and "expression" is a composition of variables and elements of the alphabet with a symbol for alternative "|."

Definition 1 Propositional Logic language

The language of propositional logic with a set Ψ of atomic formulas is defined by
$\langle\text{formula}\rangle ::= a \mid \bot \mid \neg\langle\text{formula}\rangle \mid \langle\text{formula}\rangle \wedge \langle\text{formula}\rangle$
where $a \in \Psi$.

We denote by Φ is the set of formulas (strings generated by the above grammar).

Definition 2 Modal Logic language

The syntax of modal logic includes the symbol $\Box$ and is defined by
$\langle\text{formula}\rangle ::= a \mid \bot \mid \neg\langle\text{formula}\rangle \mid \langle\text{formula}\rangle \wedge \langle\text{formula}\rangle \mid \Box\langle\text{formula}\rangle$
and $\Box\varphi$ may be read as "it is necessarily true that φ."

2.2 Probabilistic and statistical definitions

The definitions in this section follow the presented in the work of James (2006).

Definition 3 Random experiment

A random experiment is a uncontrolled and reproducible observation in a finite set of possible observable results.

Definition 4 Sample space

Given a random experiment, the sample space, named Ω, is the set of all possible observable results.

Definition 5 Event

Let w, with $\omega \subseteq \Omega$ be a set denoting an event of Ω. If ω is a singleton set, ω is elementary. If $\omega = \Omega$, then $\Pr(\Omega) = 1$ (the probability of Ω is 1) and ω is certain; if $\omega = \emptyset$ it is an impossible event, $\Pr(\emptyset) = 0$.

Definition 6 Real random variable

A real random variable X is a function $X\colon \Omega \to \mathbb{R}$ that maps the results of a random experiment in $\mathbb{R}$ where Ω is the sample space.

From now on when we say random variable it is a real random variable.

Definition 7 Occurrence of a random variable

An occurrence of a random variable $X\colon \Omega \to \mathbb{R}$ is $x \in \mathbb{R}$, the result of the random experiment that X maps.

A random variable is characterised by how its values are distributed: its distribution. Knowing the distribution of a random variable X is equivalent to be able to compute the probability of belonging itself in a sample space $\mathcal{A}$.

Definition 8 Accumulated distribution function

A function $F(t) = \Pr(X \leq t), t \in \mathbb{R}$ characterises a probability distribution, namely it is an accumulated distribution function, if it satisfies the following properties.

(i) *It is a non-decreasing function: for all $t_1 < t_2$, $F(t_1) \leq F(t_2)$.*

(ii) *It is continuous to the right: if $t_n \downarrow t$ (t_n as a limit ant for t) where $n \to \infty$ then $F(t_n) \to F(t)$;*

(iii) $\lim_{t\to-\infty} F(t) = 0$ *and* $\lim_{t\to\infty} F(t) = 1$.

Definition 9 Continuous random variable

Given a random variable X and its accumulated distribution function $F(t)$, the probability density function of X is $f(t) = \partial F(t)/\partial t$, that defines a continuous random variable if $f(t)$ exists, satisfying the following properties.

(i) It is always positive: $f(t) \geq 0$.

(ii) Its integral in $\mathbb{R}$ is 1, so $\int_{\mathbb{R}} f(t)dt = 1$.

In this work we use only continuous random variables. So, when referring only to random variables it means continuous random variables.

Definition 10 Random variables with joint distribution

In a random experiment where the interest leads to the joint behaviour of two or more random variables, namely $(X_1, X_2, \ldots, X_n)$, the probability density function is characterised by a function $f_{X_1,X_2,\ldots,X_n}(t_1, t_2, \ldots, t_n)$ where

(i) $f_{X_1,X_2,\ldots,X_n}$ is not negative;

(ii) $\int \cdots \int f_{X_1,X_2,\ldots,X_n}(t_1, t_2, \ldots, t_n)dt_1dt_2 \ldots dt_n = 1$.

Definition 11 Expectation of a random variable

The expectation of a random variable X is a weighted average where the weights are given by the probabilities $\Pr(X = t)$. It is equivalent to $\mathbb{E}[X] = \int_{\mathbb{R}} tf(t)dt$, if this integral exists.

Definition 12 Expectation of a function

Given X a random variable and $\psi(X)\colon \mathbb{R} \to \mathbb{R}$ a function, the expectation of $\psi(X)$ is $\mathbb{E}[\psi(X)] = \int_{\mathbb{R}} \psi(t)f(t)dt$, if this integral exists.

Definition 13 Conditional probability

Given X and Y random variables. The conditional probability $\Pr(X \leq x \mid Y = y)$ is defined as

$$\frac{\Pr(X \leq x \text{ and } Y = y)}{\Pr(Y = y)}.$$

Definition 14 Conditional expectation

The conditional expectation is defined as $\mathbb{E}[X \mid Y] = \int_{\mathbb{R}} xf(x \mid Y = y)dx$. If X and Y have a joint distribution, then

$$f(x \mid Y = y) = \frac{f_{XY}(x, y)}{f_Y(y)}.$$

Definition 15 Exponential distribution

A random variable X follows an exponential distribution $e(\lambda)$ of rate λ if its density function is

$$f(t \mid \lambda) = \begin{cases} \lambda e^{-\lambda t} & \text{if } t \geq 0 \\ 0 & \text{if } t < 0. \end{cases}$$

such that $\lambda \in \Theta$, where $\Theta = \mathbb{R}_+$ is the parametric space and $t \in \mathbb{R}$.

Definition 16 Stochastic process

A Stochastic Process $\{S(x)\colon x \in \mathbb{R}^d\}, d \geq 1$ is a collection of random variables where $x \in \mathbb{R}^d$ is the index of each random variable $S(\cdot)$.

Definition 17 Markov Chain

A Markov Chain is a stochastic process $\{S(x)\colon x \in \mathbb{R}^d\}, d \geq 1$ where the conditional probability distribution of any future states of $S(x)$ depends only on the actual state. The process is called "memoryless" (positional).

Definition 18 Continuous Time Markov Chain (CTMC)

A Continuous Time Markov Chain is a Markov Chain $\{S(x)\colon x \in \mathbb{R}^d\}, d \geq 1$ such that the distribution of the collection of random variables is continuous.

2.3 Propositional Dynamic Logic and Petri Nets

This section presents a brief overview of two topics on which the development of this thesis is based on. First, we make a brief review of the syntax and semantics of PDL (Harel et al., 2000; Harel, 1980; Fischer & Ladner, 1979). Second, we present the Petri Nets formalism and its variant, Marked Petri Nets; hence, the Petri Nets approach used in this work (de Almeida & Haeusler, 1999) is presented.

2.3.1
Propositional Dynamic Logic

In this section, we present the syntax and semantics of PDL.

Definition 19 PDL language

The PDL language consists of a set Φ *of countably many proposition symbols, a set* Π *of countably many basic programs, the boolean connectives* $\neg$ *and* $\wedge$*, the program constructors* $;$, $\cup$ *and* $\star$ *and a modality* $\langle\pi\rangle$ *for every program* π*. The formulas are defined as follows:*

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \langle\pi\rangle\varphi, \ \textit{with} \ \pi ::= a \mid \pi_1;\pi_2 \mid \pi_1 \cup \pi_2 \mid \pi^\star,$$

where $p \in \Phi$ *and* $a \in \Pi$*.*

In all the logics that appear in this paper, we use the standard abbreviations $\bot \equiv \neg\top$, $\varphi \vee \phi \equiv \neg(\neg\varphi \wedge \neg\phi)$, $\varphi \to \phi \equiv \neg(\varphi \wedge \neg\phi)$ and $[\pi]\varphi \equiv \neg\langle\pi\rangle\neg\varphi$.

Definition 20 PDL frame

A frame *for PDL is a tuple* $\mathcal{F} = \langle W, R_a\rangle$ *where*

- W *is a non-empty set of states;*
- R_a *is a binary relation over* W*, for each basic program* $a \in \Pi$*;*
- *We can inductively define a binary relation* R_π*, for each non-basic program* π*, as follows*
  - $R_{\pi_1;\pi_2} = R_{\pi_1} \circ R_{\pi_2}$,
  - $R_{\pi_1 \cup \pi_2} = R_{\pi_1} \cup R_{\pi_2}$,
  - $R_{\pi^\star} = R_\pi^\star$, *where* $R_\pi^\star$ *denotes the reflexive transitive closure of* R_π*.*

Definition 21 PDL model

A model *for PDL is a pair* $\mathcal{M} = \langle\mathcal{F}, \mathbf{V}\rangle$*, where* $\mathcal{F}$ *is a Petri-PDL frame and* $\mathbf{V}$ *is a valuation function* $\mathbf{V} : \Phi \to 2^W$*.*

The semantical notion of satisfaction for PDL is defined as follows:

Definition 22 *PDL satisfaction notion*

Let $\mathcal{M} = \langle\mathcal{F}, \mathbf{V}\rangle$ *be a model. The notion of* satisfaction *of a formula* φ *in a model* $\mathcal{M}$ *at a world* w*, notation* $\mathcal{M}, w \Vdash \varphi$*, can be inductively defined as follows:*

- $\mathcal{M}, w \Vdash p$ *iff* $w \in \mathbf{V}(p)$*;*
- $\mathcal{M}, w \Vdash \top$ *always;*

- $\mathcal{M}, w \Vdash \neg\varphi$ *iff* $\mathcal{M}, w \nVdash \varphi$;
- $\mathcal{M}, w \Vdash \varphi_1 \wedge \varphi_2$ *iff* $\mathcal{M}, w \Vdash \varphi_1$ *and* $\mathcal{M}, w \Vdash \varphi_2$;
- $\mathcal{M}, w \Vdash \langle\pi\rangle\varphi$ *iff there is* $w' \in W$ *such that* $wR_\pi w'$ *and* $\mathcal{M}, w' \Vdash \varphi$.

PDL has a Tableaux-based deductive system (De Giacomo & Massacci, 1998) with a web-based implementation (Schmidt, 2004). Notice that First Order PDL is undecidable (Harel et al., 2000) and that its SAT problem is EXPTime-Complete (proved by a reduction of the Two-Person Corridor Tiling problem – Blackburn et al., 2001). As usage examples we have the following.

Algorithm 1 receives as in input two boolean values A and B and computes a boolean expression. This program can be modelled as $[(C \longleftarrow \neg A \vee B); (\neg B) \cup (\neg A)]w$ where w will be the boolean value returned. This formula is equivalent to $[(C \longleftarrow \neg A \vee B)][(\neg B)]w \wedge [(C \longleftarrow \neg A \vee B)][(\neg A)]w$.

```
Data: Two boolean values $A$ and $B$
Result: A boolean value
$C \longleftarrow \neg A \vee B$;
if $\neg C \vee (A \wedge B)$ then
    $\neg B$;
else
    $\neg A$;
end
```

Algorithm 1: Evaluates a boolean expression

Some programs that may be expressed as first order formulas can be encoded in PDL. Concerning algorithm 2 that receive as input three numbers and computes the average of the two smallests and sum this result with the biggest.

Supposing p as a propositional symbol that means that "the program stopped", verify if this program stops is equivalent to verify if the formula $[((m \longleftarrow (y+z)/2); (r \longleftarrow m+x)) \cup ((m \longleftarrow (x+z)/2); (r \longleftarrow m+y)) \cup ((m \longleftarrow (x+y)/2); (r \longleftarrow m+z))]p$ is valid. Notice that it is equivalent to compute if $[((m \longleftarrow (y+z)/2); (r \longleftarrow m+x))]p \wedge [((m \longleftarrow (x+z)/2); (r \longleftarrow m+y))]p \wedge [((m \longleftarrow (x+y)/2); (r \longleftarrow m+z))]p$ or to compute if $[((m \longleftarrow (y+z)/2)][(r \longleftarrow m+x))]p \wedge [((m \longleftarrow (x+z)/2)][(r \longleftarrow m+y))]p \wedge [((m \longleftarrow (x+y)/2)][(r \longleftarrow m+z))]p$ are valid.

Supposing q a propositional symbol that some property desired to verify if it is true after the running of algorithm 3, it is equivalent to verify if the formula $[(r \longleftarrow 1); (r \longleftarrow r \times x)^\star; (r \longleftarrow r+1)]q$ is true. It is also equivalent to compute if the formula $[(r \longleftarrow 1)][(r \longleftarrow r \times x)^\star][(r \longleftarrow r+1)]q$ is true.

```
**Data**: Three real numbers $x$, $y$ and $z$
**Result**: Average of the two smallests plus the biggest
**if** $x \geq y$ *and* $x \geq z$ **then**
    $m \longleftarrow (y + z)/2$;
    $r \longleftarrow m + x$;
**else**
    **if** $y \geq x$ *and* $y \geq z$ **then**
        $m \longleftarrow (x + z)/2$;
        $r \longleftarrow m + y$;
    **else**
        $m \longleftarrow (x + y)/2$;
        $r \longleftarrow m + z$;
    **end**
**end**
```

Algorithm 2: Computes the average of two numbers in three plus the biggest

```
**Data**: Two integer numbers $x$ and $y$
**Result**: $x^y + 1$
$r \longleftarrow 1$;
**for** $i = 1$ **to** $y$ **do**
    $r \longleftarrow r \times x$;
**end**
$r \longleftarrow r + 1$;
```

Algorithm 3: Computes the successor of the power of a number by another

2.3.2 Petri Nets

A Petri Net (Petri, 1962) is a 3-tuple $\mathcal{P} = \langle P, T, L \rangle$, where P is a finite set of places, T is a finite set of transitions with $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$ and L is a function which defines directed edges between places and transitions and assigns a $w \in \mathbb{N}$ that represents a multiplicative weight for the transition, as $L\colon (P \times T) \cup (T \times P) \to \mathbb{N}$ (in this work we consider all $w = 1$).

Marked Petri Nets

A Marked Petri Net is a 4-tuple $\mathcal{P} = \langle P, T, L, M_0 \rangle$ as above but for M_0 as an initial markup (the amount of tokens distributed in each place). In this work when referring to a Petri Net it is a Marked Petri Net.

The flow of a Petri Net is defined by a relation $F = \{(x, y) \mid L(x, y) > 0\}$. Let $s \in P$ and $t \in T$. The preset of t, named ${}^\bullet t$, is defined as ${}^\bullet t = \{s \in P\colon (s, t) \in F\}$; the postset of t, named $t^\bullet$ is defined as $t^\bullet = \{s \in P\colon (t, s) \in F\}$. The preset of s, named ${}^\bullet s$, is defined as ${}^\bullet s = \{t \in T\colon (t, s) \in F\}$; the postset of s, named $s^\bullet$ is defined as $s^\bullet = \{t \in T\colon (s, t) \in F\}$.

Given a markup M of a Petri Net, we say that a transition t is enabled on M if and only if $\forall x \in {}^\bullet t, M(x) \geq 1$. A new markup generated by setting a transition which is enabled is defined as

$$M_{i+1}(x) = \begin{cases} M_i(x) - 1, & \forall x \in {}^\bullet t \setminus t^\bullet \\ M_i(x) + 1, & \forall x \in t^\bullet \setminus {}^\bullet t \\ M_i(x), & \forall x \notin \{({}^\bullet t \setminus t^\bullet) \cup (t^\bullet \setminus {}^\bullet t)\} \end{cases} . \tag{2-1}$$

A program behaviour is described by the set $M = \{M_0, \ldots, M_n\}$ of a Petri Net markups.

A Petri Net may be seen in a graphical representation, using a circle to represent each $s \in P$, a rectangle to represent each $t \in T$, the relations defined by L as edges between places and transitions. The amount of markups from M are represented as filled circles into the correspondent places. An example of a valid Petri Net is in Figure 2.1.

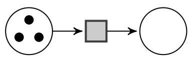


Figure 2.1: Example of a valid Petri Net

Just as an example, the Petri Net on Figure 2.2 represents the operation of an elevator for a building with five floors. A token in the place U indicates that the elevator is able to go up one floor; and, when T_1 fires, a token goes to D, so the elevator can go down a floor. If the elevator goes down a floor (i.e. T_2 fires) a token goes to the place U. Figure 2.2 illustrates the Petri Net with its initial markup.

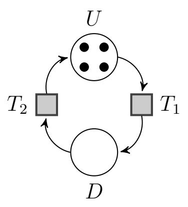


Figure 2.2: Petri Net for a simple elevator of five floors

Another example is in Figure 2.3, which represents a SMS send and receive of two cellphones. When the user sends a SMS from his cellphone, it goes to his phone buffer (i.e. T_1 fires and the token goes to p_2). When the phone sends the message to the operator (i.e. T_2 fires) it goes to the operator buffer; so, the messages must be sent to the receiver, but the receiver is able to receive only one message at a time. If there is a message in the operator buffer and the receiver is not receiving other message (i.e. there is at least a token in p_3 and there is a token in p_4), the receiver can receive the message (i.e. T_3 fires). At this point the user cannot receive other messages (i.e. there is no token in p_4, so T_3 is not enabled); but, after the complete receive of the message (i.e. T_4 fires), the receiver is able to receive messages again (i.e. there is a token in p_4 and when p_3 have at least a token, T_3 will be enabled again).

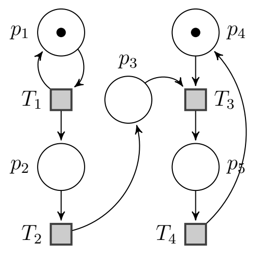
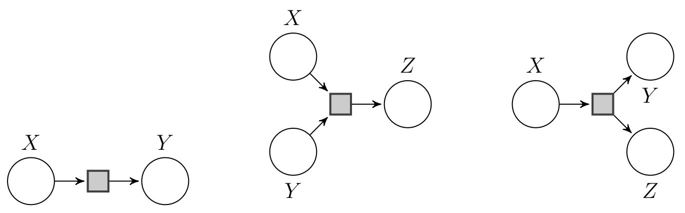


Figure 2.3: Petri Net for a SMS send and receive

Basic Petri Nets

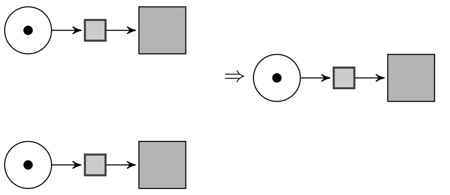
The Petri Net model used in this work is as defined by de Almeida and Haeusler (de Almeida & Haeusler, 1999). It uses three basic Petri Nets which define all valid Petri Nets due to its compositions. These basic Petri Nets are as in Figure 2.4.



2.4(a): Type 1: t_1 2.4(b): Type 2: t_2 2.4(c): Type 3: t_3

Figure 2.4: Basic Petri Nets

To compose more complex Petri Nets from these three basic kinds of composition, there is used a gluing procedure (de Almeida & Haeusler, 1999). The operations involved in this process are Conflict on the Left (Figure 2.5(a)), Conflict on the Right (Figure 2.5(b)), two cases of Sequence (Figures 2.6(a) and 2.6(b)), Joint on the Left (Figure 2.6(c)), Joint on the Right (Figure 2.6(d)) and three cases of Repetition (figures 2.7(a), 2.7(b) and 2.7(c)).


2.5(a): Conlict-L

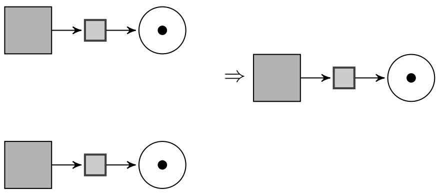
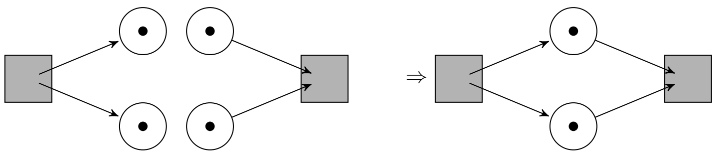
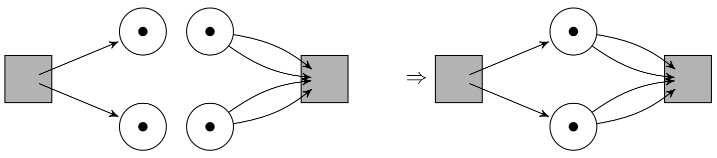
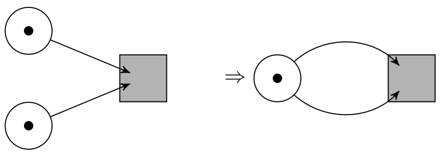

2.5(b): Conlict-R

Figure 2.5: Example of application of Conflict rules, where black boxes represent any valid subnet of a Petri Net


2.6(a): Sequence case 1


2.6(b): Sequence case 2


2.6(c): Joint-L

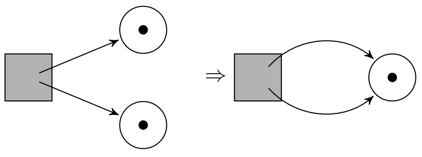
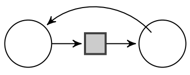
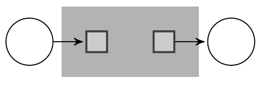

2.6(d): Joint-R

Figure 2.6: Examples of Sequence and Joint rules applications where black boxes represent any valid subnet of a Petri Net


2.7(a): Case 1


2.7(b): Case 2

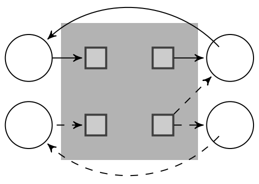
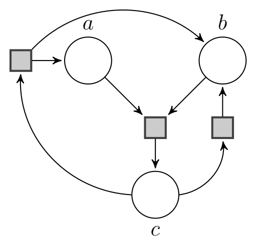
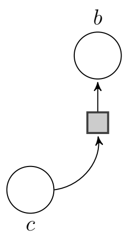

2.7(c): Case 3

Figure 2.7: Examples of the three cases of Repetition rule application

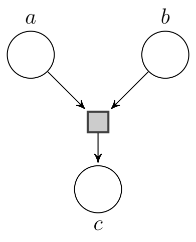
As an example, take the Petri Net of Figure 2.8(a). It is a composition of the basic Petri Nets of figures 2.8(b), 2.8(c) and 2.8(d), where the same place names indicate that when gluing they will collapse.



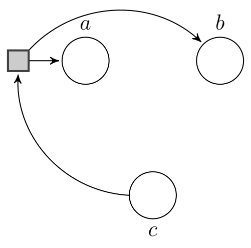
2.8(a): Composed Petri Net



2.8(b): Basic Petri Net Type 1



2.8(c): Basic Petri Net Type 2



2.8(d): Basic Petri Net Type 3

Figure 2.8: Example of Petri Net composition with its basic Petri Nets