MASTER IN COMPUTER SCIENCE

An Escape Game Theory

VERJANS, Martin

Award date:
2023

Awarding institution:
University of Namur

UNIVERSITÉ DE NAMUR
Faculté d'informatique
Année académique 2022-2023

An Escape Game Theory

Martin Verjans

.... (Signature pour approbation du dépôt - REE art. 40)

Promoteur : VANHOOF Wim

Co-promoteur : YERNAUX Gonzague

Mémoire présenté en vue de l'obtention du grade de Master 60 en Sciences Informatiques

Faculté d'Informatique – Université de Namur

RUE GRANDGAGNAGE, 21 ● B-5000 NAMUR(BELGIUM)

Acknowledgments

I wish to thanks my co-tutor, Gonzague Yernaux. This subject proposal was a bottle in the sea and looked interesting at a first glance. After a few readings, it appeared that he hit the bull's eyes. Reading an article explaining how and why your thesis' subject is an actual need is a nice feeling. I wish to thank him for his time, patience and precious advices.

I also wish to thank Simon Dourte, a student of Namur University. During one of his software development courses, he was assigned with the mission of building a functional software based on my ideas. Despite the fact that the framework was constantly evolving, he managed to produce a working piece of software. Thanks to him for he allowed me to validate or refute many ideas.

Also, a special thanks to my life companion. Her patience (and coffees) managed to keep me working during late hours. Without her approbation, I wouldn't have written anything. Thank you, I owe you a lot.

Résumé

La littérature dédiée aux Escape Games reste limitée, en dépit de leur prolifération ces dernières décennies. En particulier, l'absence d'un framework formel et bien établi élucidant la nature fondamentale des Escape Games est flagrante. Bien qu'en reconnaissant la nécéssité, la littérature scientifique n'a pas encore produit de définition formelle. Ce travail constitue un premier pas permettant de combler ce vide. Nous construisons un framework, basé sur les concepts trouvés dans la littérature existante, en général introduits de manière informelle. Le résultat produit est un premier framework fonctionnel permettant de représenter, de manipuler et de raisonner mathématiquement avec ces jeux à travers une représentation des Escape Games en tant que graphes gouvernés par des règles explicites. Nous discutons aussi des extensions possibles de ce framework.

Mots-clés *: Escape Games, Définition, Framework, Théorie des graphes*

Abstract

The body of literature dedicated to Escape Games remains limited, despite their proliferation for several decades. In particular, the absence of a well-established formal framework for elucidating the fundamental nature of Escape Games is conspicuous. While acknowledging the need for it, literature did not produce a valid formalized definition yet. This work constitutes a first step towards filling this gap. We build a framework, based on existing literary sources where the game concepts are typically introduced in an informal manner. This results in a first working framework for representing, manipulating and reasoning on a mathematical level with such games, notably through a representation of Escape Games as graphs governed by explicit rules. We also discuss extensions to this framework.

Keywords *: Escape Games, Definition, Framework, Graph Theory*

Contents

1 Introduction

Escape Games are a phenomenon that has widely developed over the last few years. From early video games to real physical rooms or buildings and more recently Virtual Reality, they are very popular amongst the young generation. Scientific literature is slowly getting interested in these new kind of games, mainly for educational purposes.

However, while many articles are interested in the realization of an Escape Game to teach a specific topic, no real definition has been posed on what is an Escape Game. Wiemker, Elumir and Clare (2015)[34] have published an article that is trying to provide an overview of Escape Games and to expose their main concepts. This will serve as a basis for this work.

The original research question aimed at providing a mathematical definition of what an Escape Game is. Game theories in mathematics include more and more games, but Escape Games still don't have one of their own.

It is worth noting that Escape Games are sometimes mentioned as “Escape the Room”, “Escape Rooms”, “Exit Games”, “Exit Rooms” with a few other variations. The choice here has been made to call them “Escape Games”, as it seems it is best suited for all platform variations.

Before deep diving into escape games, it seems relevant to figure out the growing interest on this matter. Using the search query (in Google Scholars) **"Escape Game" OR "Escape Room" OR "Exit Game" OR "Exit Room"** and filtering by year, we can observe that the publication count explodes from 2016. Using an adapted query on various search engines (ScienceDirect, Springer, ACM Digital Library and DBLP), we can observe the same tendencies with little variations. Figure 1.1[1] illustrates this tendency.

It seems the first digital Escape Game, as in “you must escape the room”, has been released in 1988 by John Wilson[15]. It was only a text-based game but already contained the main ingredients on what makes contemporary Escape Games. According to Scott Nicholson (2015)[25], it seems the first real live action Escape Game started in Japan in 2007. As of today, they exists in various and mixed forms and environments. A simple search on Google Play©yields thousands of results on Escape Games for smartphones, available for free or not. You can find books, board games, card games that are all Escape Games. Recently, Virtual Reality games have developed quite a lot and Escape Games are a part of those games too.

All those games on all those platforms must have something in common to be called Escape Games. There should be a definition, formal or not, that defines these games so that a line may be drawn as where they begin to be Escape Games and where they are no more. Surprisingly, there is no such definition yet. Nicholson (2015)[25] and Wiemker, Eljumir and Clare (2015)[34] provided us with the main concepts they have identified as constituent of an Escape Game. No real major addition could be found to these concepts.

1.1 The research question

The original idea behind this work was : “Is it possible to develop an Escape Game Theory using mathematics' Game Theory ?”. Since nobody seemed to have answered that question yet, it seems our job is to make an attempt at building a mathematical theory for Escape Games.

[1] Actual data can be found in Appendix A

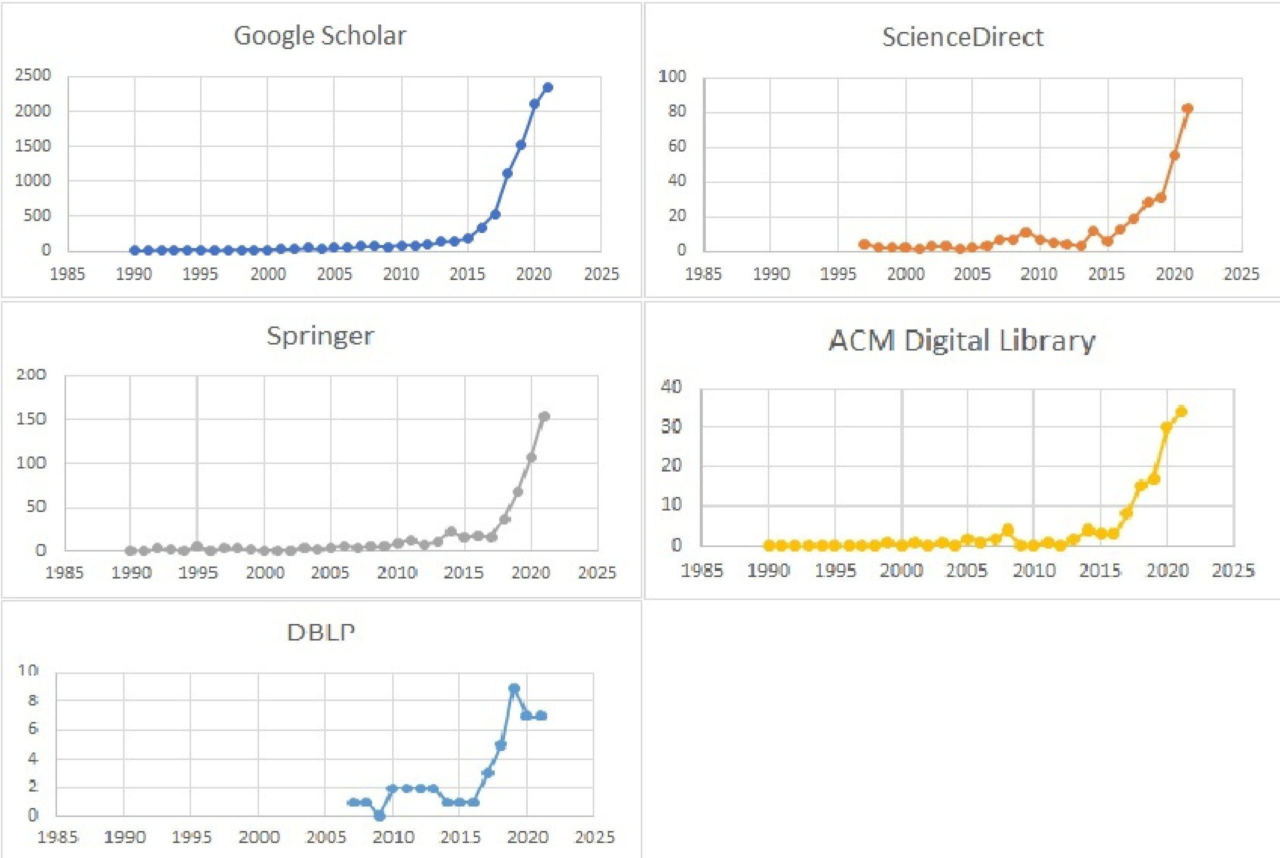


Figure 1.1: Number of articles by year on different search engines containing any of the words "Escape Game", "Escape Room", "Exit Game" or "Exit Room"

If possible, the goal will be to provide this theory. The Escape Games research field is brand new and in need of methodological methods in order to validate/measure a game :

- Is the Escape Game possible to win ?
- How is the game balanced ?
- Is it possible to measure the connection between each item and the main theme in order to have a coherence score ?
- Will the game immerse the players enough in order to trigger the flow state ?
- If the game is teaching something, is it possible to measure the long term outcome ?
- Would it be possible to differentiate a “good” Escape Game from a “bad” one ?
- Is it possible to measure the cooperation level among players ?

All these questions require some kind of metrics that would help quantify and qualify a given Escape Game.

Currently, no attempt at defining an Escape Game from a mathematical point of view has been performed yet[2]. There is no article or scientific literature[3] providing an exhaustive list of games included in Game Theory. When searching using “State of the art” and “Game Theory” keywords, mainly articles are focused on whether Game Theory is really useful in real life situations[4] or provide a new field in which a particular game could be applied in order to build a model.

In last resort, the only place where it is possible to find the most complete compilation on what has been published, on which games have a theory and on the list of these games is Wikipedia[2] [3]. Obviously not belonging to scientific literature, these articles can provide an interesting starting point in order to answer the question : “Is there a Game Theory that could match or define Escape Games ?”. The short answer is no.

A previously stated, no literature (so far) has made a proper link to the mathematics field in order to formally define an Escape Game. If defined properly, a mathematical model could probably :

- decide whether a game is possible to win
- measure the complexity balance for each challenge and for the overall game

A link to computer sciences could be done. Provided that we have a mathematical model for Escape Games, it would be possible to write a program helping designers in many aspects of the design process. It could provide warnings when the game becomes too difficult or too easy, graphically assist designers as they build the rooms or pretest the game without any real item and before the room is actually built[5].

Our research then becomes : “Would it be possible to provide an escape game mathematical definition that includes the main aspects of an Escape Game and provide its evolution rules” ?

This work focuses on the realization of a formal framework capable of modelling Escape Games. We will first conduct a state of the art, then extract the main concepts out of it. We will then select concepts that are relevant to the game mechanics. Those concepts will be defined from a mathematical perspective using Graph Theory. We will introduce and define three graphs :

1. The Escape Game Graph, presenting an Escape Game as a blueprint.
2. The Game Session Frame Graph, presenting a Game “in the action” at a given moment in time.
3. The Game Session Forest, presenting how the Frames can be linked together by players’ actions and system’s reactions.

After defining those graphs, we will discuss about the framework completeness, its usability and its possible future extensions. We shall then conclude this work.

[2]At least, none that the author could find.
[3]Again, not that the author could find.
[4]For example, Guerrien, Bernard (2018)[16]
[5]Although this may seem less relevant, the industry world is usually enthusiastic when provided with a cost saving tool.

2 State of the art

2.1 Article classification

The literature on Escape Game is vast, as shown in Figure 1.1. Therefore we choose to sort the vast amount of articles into several categories :

1. Taxonomy;
2. The realization/implementation of an Escape Game, usually for educational purposes, with validation using student sets;
3. The realization of an item or device to be used in Escape Games by teachers;
4. States of the art, history and surveys;
5. Escape Games examined through the collaborative and teamwork lens.
6. Design frameworks, proposed to help designers create great games.

2.1.1 Taxonomy

Recently, a taxonomy on puzzles encountered in Escape Games has been developed[22]. The authors define a puzzle as a set of atomic challenges[1] that have to be solved in order to solve the puzzle[2]. They break the challenges into three main categories :

1. Mental challenges, where the players have to "use their brains"
2. Physical challenges, where a physical action in the real world (or the gaming environment) is required
3. Emotional challenges, where players have to deal with their internal emotions (fear, disgust...)

Since this is the very first taxonomy on Escape Game puzzles (or in the Escape Games field), it is the only article in this category.

2.1.2 Escape Game implementations

By far the most populated category. It is about the creation and validation of an Escape Game aimed at teaching a specific subject for a specific student set. The structure of these articles is usually the same :

- Introducing the matter to learn
- Introducing the target audience (student profiles)
- Designing the Game
- Evaluating the Game

[1] This article is the reason the author choose to use the word Challenge instead of Puzzle.

[2] This definition is really convenient for computer science and mathematics as the challenges are atomic, therefore a complete taxonomy of atomic challenges would allow to model any puzzle.

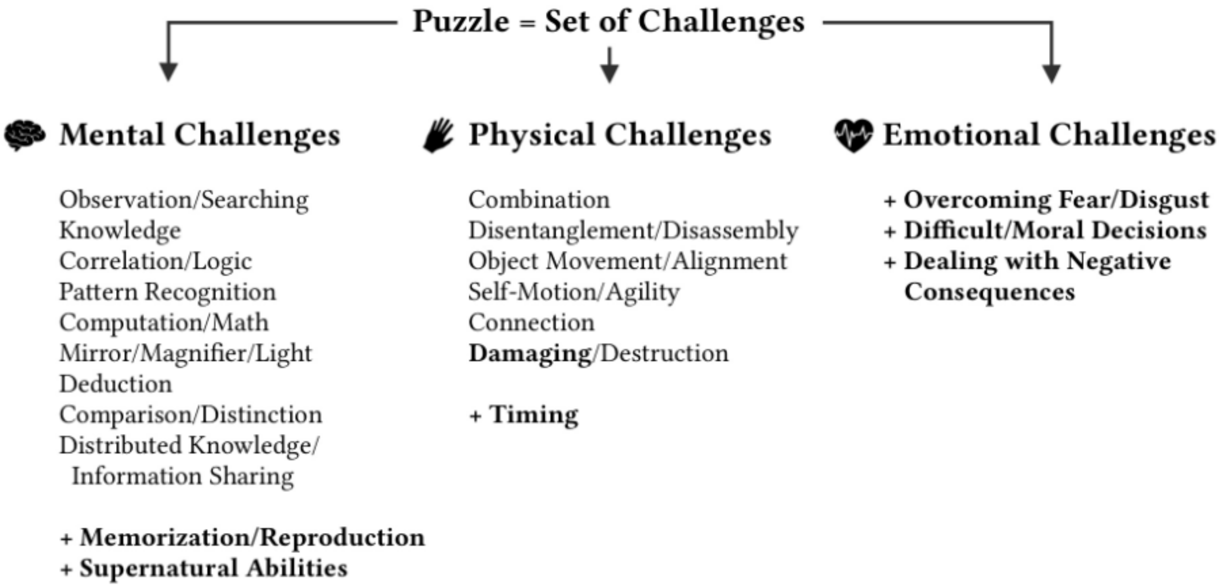


Figure 2.1: Puzzle taxonomy, from Krekhov, Andrey and Emmerich, Katharina and Rotthaler, Ronja and Krueger, Jens (2021)

In “Escape Rooms as a Way to Teach Magnitudes and Measure in Degrees in Education”[5], Students in primary schools discover the notions of mass, volumes and lengths. “An escape room as a simulation teaching strategy”[7] provides a course for nursing student. They must analyse patients’ data in order to find clues that will help them open the locks. “Designing an escape room game to develop problem solving and spatial reasoning skills”[11] proposes an escape game to develop problem solving as well as spatial reasoning skills. “Escaping the routine: Unlocking group intervention”[12] is about making students understand the principle of diverse approaches from group members and how interdependence leads to positive group outcomes[3]. “The use of escape rooms to teach and learn English at university”[23] is aimed at helping Spanish students learn English in a unconventional way. Ho, Anne M (2018)[19] developed an Escape Game aimed at understanding cryptography.

The main issue with these studies is that the Escape Games are often self-validated and the outcomes cannot be easily calculated. The question : “Did these student learn better from an Escape Game than from traditional teaching ?” cannot be answered straightforward. Taraldsen and all (2020)[31] perform an analysis and a systematic review on Escape Games for educational purposes implementation. They state that :“[..] it seems that the research on the use of escape rooms for educational purposes has reached a new phase, which requires greater structure and transparency in research design and methods for data collection and analysis.”

This sentence is actually validating the need for this work. Providing a transparent, standard and formal framework is a first step towards a global analysis of educational escape games.

2.1.3 Device to be used in Escape Games

We shall not deep dive into this section, as not really relevant for our research question. It is worth noting that this kind of work exists in order to help teachers build Escape Games at low costs. Ross, Robert (2019)[30], for example, created a small numerical keypad-based lock, equipped with a timer that can be built by anyone for $30.00 AUD.

2.1.4 States of the art, history and surveys

Nicholson (2015)[25] conducts a survey on 175 Escape Game facilities. It is the first survey we could find and it explores demographics, facility descriptions, costs, themes and narratives, puzzle organisation...

[3]from the abstract

Penttilä, Katriina (2018)[29] provides a "History on Escape Games" as a master thesis. The author states that Dr. Nicholson's white paper "Peeking behind the locked door: A survey of escape room facilities"[25] is the first real article addressing the Escape Game field. A history of Escape Games from their premises until the late 2010s is analysed.

Fotaris, Panagiotis and Mastoras, Theodoros (2019)[14] conduct a systematic review on educational Escape Games.

2.1.5 Collaboration

Escape Games are essentially collaborative games[4]. Players must overcome challenges as a team. "Collaboration, Awareness, and Communication in Real-Life Escape Rooms"[27] conduct an observational study in order to understand the collaborative process in Escape Games. "Escape Rooms as a Collaborative Problem-Solving Environment"[28] follows groups of students among an Escape Game, enlightens the collaborative process as well as the meta-cognition[5] process that takes place during the game.

2.1.6 Design Frameworks

A few design frameworks have been developed[6] in order to aid designers create Escape Games. To name a few :

- Ask why ?[26]
- The Experience Pyramid Model[18]
- EscapED [10]
- The Star Model[6]

Nicholson (2016) states that a properly designed Escape Game should contain only challenges and items that have a connection to the main story and theme. This will bring consistency throughout the game and allow better immersion for the players. It is less of a framework than a series of guidelines, intended to enlighten designers on how to create better games.

Heikkinen, Outi and Shumeyko, Julia (2016) propose a Framework, the Experience Pyramid Model and build an Escape Game with it. Even if they don't validate the game with real players, they claim that this model is well suited for Escape Game design. They explain the model, originally introduced by Tarssanen (2009)[32], which consist in 5 levels of perception. From Tarssanen (2009) : "The model represents an ideal product, which takes into account all the elements at all levels of a meaningful experience. It is an explicit tool for identifying critical points or deficiencies in the product, and thus, for designing the product further."

The EscapED theoretical framework is a 6 steps guide in order to design a proper Escape Game. Its main goal is to assist Escape Game designers. It helps defining the boundaries and challenges of the game in the proper order, which is :

1. Participants
2. Objectives
3. Theme
4. Puzzles
5. Equipment
6. Evaluation

The authors provided the framework to designers and analysed how they created games based on it. Although no formal study has been conducted, the feedback was rather positive. However they noticed that the Theme step was usually set back by the designers and usually took place amongst the last steps of the design process; they concluded by stating that further studies were required in order to properly define this particular step position.

[4]Although some platforms do not allow to play them as a team (smartphones, VR devices...), it is likely that multi-player versions of these games will be developed in the near future.

[5]They do so by using Chiu, Ming Ming and Kuo, Sze Wing (2009)[8] meta-cognition framework.

[6]Some books also provide design guidelines or frameworks, "Escape The Game: How to make puzzles and escape rooms"[9], "Games, puzzles, and computation"[17], "Theory of fun for game design"[21] for example.

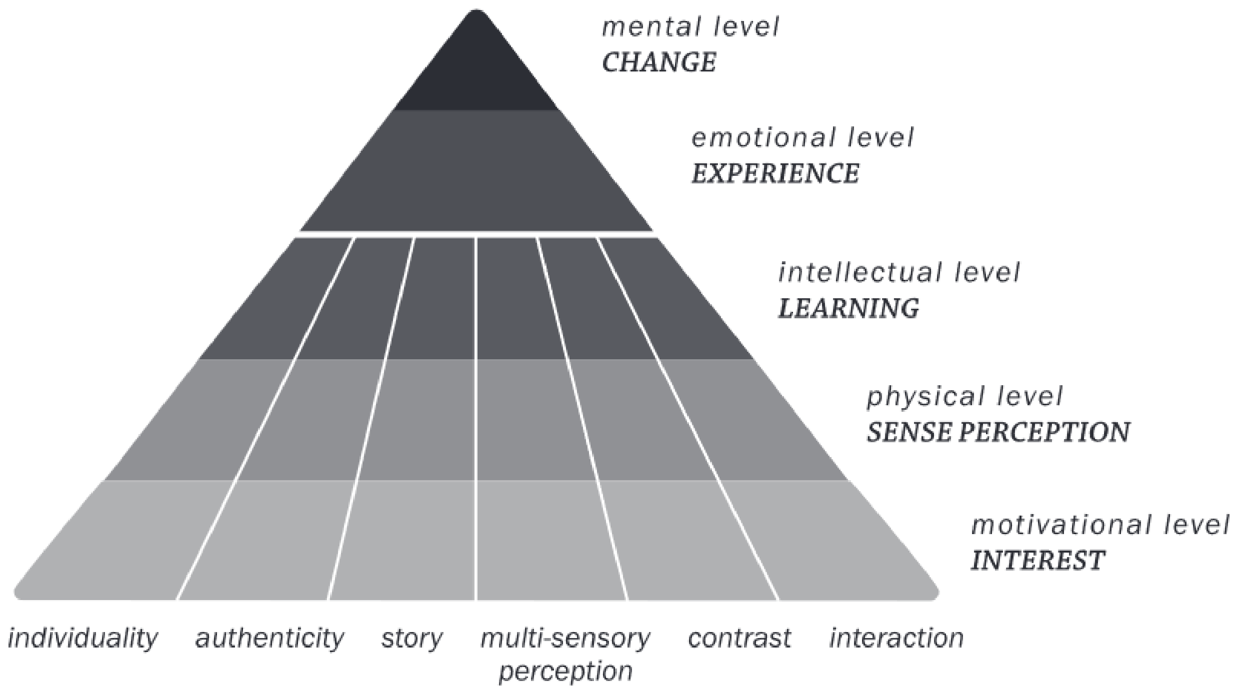


Figure 2.2: The Experience Pyramid Model, from Tarssanen (2009)

Figure 2.3: The EscapED framework, from Clarke, Samantha and Peel, Daryl and Arnab, Sylvester and Morini, Luca and Keegan, Helen and Wood, Oliver (2017)

Figure 2.4: The Star Model, from Botturi, Luca and Babazadeh, Masiar (2020)

More recently, Botturi, Luca and Babazadeh, Masiar (2020)[6] developed a design framework, the Star Model, in order to help teachers create better Escape Games. This framework is mainly intended for educational purposes. It is a two layers model, containing as first layer (the Game layer) the five dimensions required to design Escape Games : Narrative, Equipment, Puzzles, Game-flow and Learning[7] The second layer (the Context layer) contains four items related to the context where the Escape Game should take place : Players, Constraints, Evaluation and Debriefing. In order to validate the model, they provided a course for teachers to attend and to evaluate on the clarity, completeness and utility of the presented model. They received an acceptable overall score in order to validate the Star Model. However, they did not evaluate any of the Games that have been produced using this model.

2.2 What is an escape game ?

We will now make an attempt at defining the main concepts that are behind the words "Escape Game". Words in **bold** are the concepts that have been discovered and will require a proper definition.

2.2.1 Unravelling main concepts from literature

Nicholson (2015)[25] defines an escape game as : "Live-action team-based games where players discover clues, solve puzzles, and accomplish tasks in one or more rooms in order to accomplish a specific goal (usually escaping from the room) in a limited amount of time". It seems to be the first definition of what an Escape Game is and some of the main concepts are taken into acount :

- **Players** are seen as a team, therefore Escape Games are above all cooperative games. Also, there are no enemies[8].
- **Clues, Puzzles, Tasks** are what is to be done in order to win the game.
- **Rooms(s)** indicate that there is a gaming environment from which the players must escape from.
- **Goal, Exit** is where the players need to go to win.
- **Time limit** is setting a constraint on victory. The Goal must be reached before the time limit ends.

Although interesting, this definition is not accurate enough as Escape Games have evolved on many platforms.

[7] This last item introduces the learning dimension and could be left over if designing non educational Escape Games.

[8] Some Escape Game companies appear to have teams play against each other, but this is marginal and doesn't really count as enemy. The other team is seen as a competitor

Live-action is not always true, as it is possible to play Escape Game on Virtual Reality devices, Smartphones, computers...

Team-based is correct for live-action games, while for digital devices it is usually one player games. Escape Game books can be played alone or together.

The concept of time limit is not perceived the same way depending on the platform. In a real live-action Escape Game, you usually have a timer indicating how much time is left. If the team doesn't find the exit in time, the game is lost. Using an Escape Game book[9], you have to monitor the time yourself, books usually provide rating at the end depending on the time you took. Some Escape Games on smartphones only monitor time in order to create ratings among the players. Some early Flash games on computers simply don't mention time.

Wiemker, Eljumir and Clare (2015) elaborate on the puzzle concept and introduce the Game Master and theme concepts :

- Puzzles;
  - **Puzzle path**;
  - **Hints**;
  - **Skills**;
- **Game Master**;
- **Theme**.

Mainly they focus on the fact that an Escape Game is above all a set of puzzles, a series of challenges that players have to overcome in order to win together against the game. They refine the concept of puzzles by explaining how the puzzles are arranged altogether, as the reward for solving one puzzle gives a clue that will allow one to solve the next puzzle. They subdivide puzzles into three main categories : Standalone puzzles, Meta-puzzle (the ultimate puzzle to open the last door) and discovery puzzle (that explains the game to the players and starts the narrative).

They add the notion of skills (what is required for the player(s) in order to solve the puzzles).

They introduce the concepts of Game Master, a person that is here to help the players (should they get stuck) by giving them hints.

Finally they enlighten the concept of theme, which is the story that is built around the puzzle path in order to immerse the players and make them enter the state of flow (see further).

Nicholson (2016)[26] provides information on how to properly design an Escape Game and mention the fact that all elements inside the room must be connected to the main theme in order to bring **coherence**. Using the method "Ask why?"[10], he helps designers create immersive Escape Games.

In their conference paper[27], Pan, Lo and Neustaedter (2017) state that "[..] escape rooms can provide **learning environments** for exploring social questions around leadership, group hierarchy, conflict, proxemics, and distributed cognition.". Escape Games can be seen a set of **collaborative problem solving tasks**, according to Papadopoulus and Tenta (2021)[28].

Last, but not least, is the concept of **flow**, originally introduced by Csíkszentmihályi (1996)[13]. Many other articles[11] reference this concept. The idea is that when a game is well designed, the player experiences a state of flow, where the immersion is so strong that performances are at their best. Although there doesn't exist a quantitative way of knowing if a given Escape Game can induce players into flow state, the aim is to provide problems that are balanced between feasibility and challenge.

[9] In "échappe-toi de la tour aux dragons"[24], every page contains a puzzle, the reward for each puzle is one symbol to unlock the final page

[10] For each element/artefact in the room, you must ask "Why is it here?", "Is the place of this object relevant to the storyline?"

[11] Hsieh, Ya-Hui and Lin, Yi-Chun and Hou, Huei-Tse (2013)[20], Veldkamp, Alice and Niese, Johanna and Heuvelmans, Martijn and Knippels, Marie-Christine and van Joolingen, Wouter (2022)[33], Admiraal, Wilfried and Huizenga, Jantina and Akkerman, Sanne and Dam, Geert (2011)[4] among others

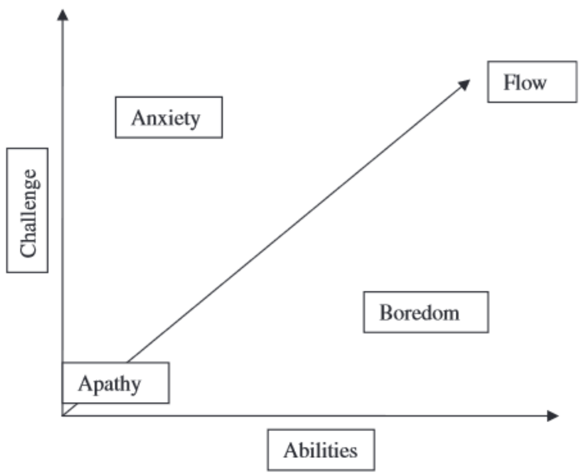


Figure 2.5: Model of flow (from Admiraal, Wilfried and Huizenga, Jantina and Akkerman, Sanne and Dam, Geert (2011), adapted from Csikszentmihalyi (1990, 1997)

3 The Escape Game Framework

In this work, we will formally define Escape Game based on the selected concepts that have been extracted from literature. We will explain why graphs have been chosen in order to define Escape Game. The Escape Game Graph will be formally defined. From it, we will derive the Game Session Frame Graph and explain the rules for defining player's actions. When the action's reactions will have been defined, the Game Session Graph will be introduced. Finally, the Game Session Forest will close the Framework definition.

3.1 Main concepts

3.1.1 A definition

Using the concepts highlighted in the introduction, we can provide our own definition of what an Escape Game is. First of all, we must define the main ideas :

Definition 3.1.1 (Player). *A player is a human being playing the game. He/She is volunteer to play and possesses skills that will help him/her overcome the challenges and win the game.*

Definition 3.1.2 (Team). *A team gathers all the players that agreed to play the game. They all loose or win together and must collaborate in order to succeed.*

Definition 3.1.3 (Skill). *A skill is a capability that players possess. It allows them to solve puzzles.*

Definition 3.1.4 (Gaming Environment). *The gaming environment is the physical, digital or imaginary*[1] *area in which the team has to evolve. The game usually ends when the whole team is out of the environment.*

Definition 3.1.5 (Challenge). *A challenge can be seen as a puzzle, task or problem that requires to be solved in order to progress in the game. It is possible to solve it by using clues, player skills and knowledge, or hints provided by the Game Master.*

Definition 3.1.6 (Game Master). *The Game Master is an entity that exists in order to help the players, should they get stuck. Depending on the platform, this entity can be human or software and can provide clues in order to help the players through the game.*

Definition 3.1.7 (Theme). *The theme is the general atmosphere during the game*[2]*.*

Definition 3.1.8 (Story (or narrative)). *The story is an interactive scenario between the game designer and the players. It is provided to the players, who are the main actors, and will unravel as they overcome challenges. The story is related to the theme.*

Definition 3.1.9 (Escape Game). *An Escape Game is a game where a player or a team of players must escape their gaming environment by solving challenges linked together by a narrative in a given time frame.*

Note that the notion of learning environment has been voluntary left behind. This is because not all Escape Games are trying to teach something and because it is still unclear whether Escape Games provide a learning environment that is better than the traditional ones.

[1] When playing a card Escape Game for instance

[2] Theme examples are : medieval, find the murderer, cyberpunk...

3.1.2 Selected concepts

From the provided definitions, we will now select which ones are to be formally represented and which one will be left behind.

Players are the central concept of Escape Games. Without players there would be no game, or at least no one to play. The concept of Team has been left behind, as the game mechanics itself will provide the team work. All players win the game or they all loose.

Puzzles is the word that has been chosen as it is the most common encounter in literature. Puzzles are the second main concept of Escape Games. Without them, the game becomes a simple maze from which players have to get out.

Rooms represent the physical universe of the game. Whether the game is virtual or real-life, players always evolve in an environment. While this concept is important, it may happen that an Escape Game involves only one room. Usually educational games must be built in order to be playable in a classroom, therefore constraining the game to only one room.

Clues are helper items or information for players. They will need them in order to solve puzzles and progress in the game.

Triggers are events that may happen in the game provided a specified condition is satisfied. A door may close when players enter a specific room for example.

Actions are activities players can perform while in the game. No article enumerating the possible player's actions could be found in the literature. Therefore we have to provide a possible action list.

Action list

As stated, no exhaustive possible action list could be found. So we decided to provide our own action list, in the hope it is sufficient to formally model Escape Games. The actions are only stated, rules validating whether an action may be performed and how the game should react will be defined in section 3.5 Actions and reactions.

A.1 Enter : A player can enter in the game.

A.2 Exit : A player can exit the game.

A.3 Move : A player can move from one room to another.

A.4 Search : A player can look in the room he/she is located for clues, puzzles or doors to other rooms.

A.5 Attempt : A player or a player group can try to solve a puzzle.

A.6 Inventory : A player can pick-up or drop an item. He/She can also memorize a informative clue.

A.7 Communicate : A player can give one of its items to another one; he/she can also tell another player about a clue he/she memorized.

3.1.3 Rejected concepts

The following concept are constituent of Escape Games. However, we choose not to retain them for different reasons. Mainly because while they are important to Escape Games, they do no affect the Game Mechanics.

Time may appear as essential. However it doesn't change the game mechanics. Only the player's attitude is different if time was to be removed from a real-life Escape Game. Puzzles will be solved the same way. Clues will be found the same way as well.

Game Master is not required in all Escape Games. It is only a helper in case players get stuck. The Game Master should not affect the game mechanics.

Theme is very important in order to build a quality Escape Game. However, since we want to build a mathematical model, theme is irrelevant in this work. We want to unveil the game mechanics, not the wrapping that has been made around it.

3.2 Chosen representation

In order to represent an Escape Game from a mathematical perspective, graphs have been chosen. An Escape Game can be represented as a graph, illustrating a blueprint for a specific game.

We will describe the objects that are part of this graph in the following section.

Usually graphs are represented with dots as nodes and a label nearby to indicate the node's name. For the sake of clarity, we choose a different representation. A node will be represented as a circle, its name inside the circle.

Edges leaving or arriving at a node will therefore arrive or leave the circle's perimeter border.

Graphs have multiple representation styles in mathematics. The one chosen in this work defines a graph G as :

$$G = (N, E, \phi)$$

where :

- G is the whole graph.
- N is the node set[3].
- E is the edge set.
- ϕ is the incidence function. It defines which nodes the edges are linking together.

The incidence function returns an ordered or unordered (depending on the graph type) node pair, provided that an edge had been passed as argument. Let e be an edge in E, linking nodes n_1 and n_2 (both belonging to N), then

$$\phi(e) = \{n_1, n_2\}$$

We may also define the incidence function as a mapping. If e_1 and e_2 are edges belonging to E and n_1, n_2 and n_3 are nodes belonging to N, then the incidence function can be defined as :

$$\phi = (e_1 \longrightarrow \{n_1, n_2\}, e_2 \longrightarrow \{n_1, n_3\})$$

meaning that e_1 link nodes n_1 and n_2 and e_2 provides a link between n_1 and n_3.

In this work, the node set be will split into several node types. The edge set as well. It doesn't affect the rules of Graph Theory as the different node types are simply a subset of the graph nodes.

3.3 The Escape Game Graph

As stated before, graphs will be used to model an Escape Game. Several graphs will be involved in the formal definition. The first one to be introduced is the most important, as it can be considered as the blueprint of a given Escape Game. The other graphs that will be built later on can be calculated based on this one.

[3] The word **node** has been selected over the word **vertex**.

The Escape Game Graph is an oriented connected graph, containing 6 node types, 1 edge set and an incidence function. Node types are : Rooms, Puzzles, Roles, Clues, Skills and Meta-information.

An Escape Game Graph E can be defined as :

$$E = (S, C, R, L, Z, I, W\phi)$$

where :

- S is the skill node set;
- C is the clue node set;
- R is the room node set;
- L is the role node set;
- Z is the puzzle node set;
- I is the meta-information node set;
- W is the edge set;
- ϕ is the incidence function

We shall define and detail each set.

3.3.1 Edges

Edges in this graph are simply used to connect nodes together. They are oriented and will be noted w_x, where x is the edge number and $w_x \in W$.

3.3.2 Skills

Skills are competences players need to possess in order to solve the different puzzles in the game. Each skill is represented once.

The Full Skill Set

To solve it, a puzzle requires different skills taken from the puzzle taxonomy built by Krekhov and all[22]. The taxonomy provided by this article divides a puzzle into a set of "atomic" challenges that can be of type Mental, Physical or Emotional. For each challenge presented in the taxonomy set, we will consider it a unique skill to possess in order to be able to solve the puzzle.

Since the taxonomy is supposed exhaustive, we can therefore build any puzzle as a combination of any of these challenges, in any order[4]. Therefore any puzzle can be represented as a combination of these atomic challenges. Since a skill can be mapped directly to an atomic challenge, any puzzle is a combination of skills.

Therefore, we shall define a skill as an element belonging to the set presented in table 3.1.

Observation/Searching	Knowledge
Correlation/Logic	Pattern Recognition
Computation/Math	Mirror/Magnifier/Light
Deduction	Comparison/Distinction
Distributed Knowledge/Information Sharing	Memorization/Reproduction
Supernatural Abilities	Combination
Disentanglement/Disassembly	Object Movement/Alignment
Self-Motion/Agility	Connection
Damaging/Destruction	Timing
Overcoming Fear/Disgust	Difficult/Moral Decisions
Dealing with Negative Consequences	

Table 3.1: Skill set from "Puzzles Unpuzzled: Towards a Unified Taxonomy for Analog and Digital Escape Room Games"

[4]Whenever the taxonomy will be found incomplete, adding the missing items will allow us to consider it exhaustive again.

This set shall be referred to as the Full Skill Set and noted S^F. This set is a constant through all Escape Games[5].

The Skill Set

The skill set used in an Escape Game is the set S, which is a subset of the Full Skill Set S^F.

Skills will be noted S_x, where x is the skill number and $S_x \in S \subseteq S^F$.
Skills possess no incoming edge and their outgoing edges go to Role nodes.

3.3.3 Clues

Clues are used by the players to solve puzzles. As general guideline rules about clues, they should be used for only one puzzle and it should be rather easy to figure out to what puzzle they belong to[6]. In this work, we will consider that players immediately recognize the puzzle linked to the clue. We will also consider that a clue may be destroyed (removed from the game) whenever the puzzle they provide help for is solved.
Therefore they possess only one outgoing edge to the role connected to their related puzzle.

They will be noted $C_{x.y}$, where x is the puzzle Z_x they relate to and y is the clue number in the puzzle. All clues in the game belong to the clue set $C : C_{x.y} \in C$.
Clues can be an information (a number painted on the wall) or an item that can be carried by players (a key). Therefore they possess a type, representing that fact, that can be noted $C_{x.y} = \{ClueType\}$, where $ClueType \in \{Item, Information\}$.

Graphically, information clues will have a dashed border while item clue's border will be plain.

3.3.4 Rooms

Rooms are the main containers in Escape Games. Anything (players, puzzles, clues) existing in the game is located in a room.

A room must have at least one incoming edge from another room, puzzle or role node in order to be reachable. Outgoing edges may lead to many other node types (Room[7], Puzzle, Clue, Meta-information).
Rooms will be noted R_x, where x is the room number and $R_x \in R$.

3.3.5 Puzzles

Puzzles are the main components in an Escape Game, as it is the reason players play. They are located in a room, therefore any puzzle shall possess at least one incoming edge coming from a room[8] or from another puzzle[9].

Puzzles require a minimum amount of players that need to solve it together. Therefore a puzzle will receive one incoming edge from a Role node for each required player.

In order to be solved, a puzzle requires different skills taken from the Full Skill Set S^F. To represent this, each Role node connected to the puzzle will receive an incoming edge for each required skill. This allows for complex puzzles to be represented, as sometimes a puzzle requires a player with good dexterity and another one with good observation skills.

Last, but not least, puzzles often require clues in order to be solved. Clues are items or information that are to be gathered by players to progress in the game. As already discussed, clues are not directly linked to the puzzles but to the roles connected to them.

[5] Provided that the taxonomy is complete, there will be no addition to this set.
[6] This is rule 5 of 13 Rules for Escape Room puzzle design[1], although not a scientific article, it provides guidelines generally accepted by the Escape Game enthusiasts community
[7] Two rooms connected together by an edge means that there are no obstacles between the rooms and players can freely move from one to another. This may seem useless and in a theoretical approach the two rooms could be fused together. However this situation is rather frequent in live action Escape Games as the building configuration is a constraint that cannot be bend easily.
[8] It is possible that a puzzle is accessible from two different rooms. It then would require one player in each room to solve it.
[9] in which case this puzzle is the reward for solving its predecessor

Puzzles offer a reward when solved (as stated in Escape room games[35]), otherwise they are pointless and create noise for the players. The reward is represented as an outgoing edge to the reward they point to[10]. The reward can be of type room, clue or puzzle.

Puzzles will be noted Z_x, where x is the puzzle number and $Z_x \in Z$.

3.3.6 Roles

Role nodes have two usages :

1. Represent the required roles in order to solve a puzzle
2. Locate the players' starting positions.

Solving puzzles In order to solve a puzzle, there must be at least one player for each role connected to the puzzle. Each role may require a different skill set and specific clues. Therefore they receive incoming edges from skills and clues and possess only one outgoing edge to the puzzle they relate to. They also receive an incoming edge from a room, They are noted $L_{x.y}$, where x is the puzzle Z_x they relate to and y the role number in the puzzle. Also, $L_{x.y} \in L$.

Starting positions Sometimes, players will start an Escape Game from different positions. To represent this, each specific starting position will be represented as a Role node and connected to a Room These nodes can also receive incoming edges from specific skill nodes, as a specific starting position could require specific skills[11]. These nodes will be noted L_x, where x is the Starting position number. Also, $L_x \in L$.

Because there are two role types, we can split this set in two as well : L^P contains roles linked to a puzzle, L^S contains roles representing a starting position. Naturally, $L^P \cup L^S = L$ and $L^P \cap L^S = \emptyset$.

The standard starting position is a specific role node. It is the player's starting position when all other starting positions have been filled with players. It will be noted L_0 and must always be present.

3.3.7 Meta-information

Meta-information node are nodes that allow the system to trigger events in the game. These events are conditional to any part of the game status. Whenever the conditions are fulfilled, the Action is executed and the graph changes. As an example, one could imagine an event triggering after all players are in a room to release a clue or to close a door.

Meta-information nodes will be noted I_x, where x is the Meta-information number. These nodes contain a condition, an action and a boolean value that are noted :

$$I_x = \{Condition, Action, Repeat\}$$

where :

- *Condition* is a boolean function; when it returns *true*, the event should be triggered.
- *Action* is a function that tells the system what to do when the event is triggered.
- *Repeat* is a boolean indicating whether the event will trigger only once ($Repeat = false$) or every time *Condition* evaluates to *true*.

Meta-Information nodes all belong to the set I.

Meta-Information nodes' Actions and Conditions only have a meaning when the game is played and that a Game Session is linked to an Escape Game Graph. This is described in further sections.

[10] It is also possible that rewards are multiple, then we would find multiple edges leaving the puzzle node

[11] The author once played an Escape Game where one of the player had to start in a prison cell, the Game Master having asked first who was not claustrophobic. This could be represented as a specific Role node connected to the cell Room and connected to the skill $OvercomingFear/Disgust$

Two Meta-information nodes will always be present in an Escape Game Graph : the Exit node I_0 and the Victory node I_1.

Exit Node The definition of the Exit Node is $I_0 = \{true, RemovePlayersInRoom(), true\}$. This node must be connected to at least one room. That room represent the Exit point and should not contain anything (i.e. there should be only one one-way connection to this room).

The condition being *true* as well as the *Repeat* value, this means the action is continuously checked. Any player present in a room that is directly linked to the Exit node will be removed from the game.

Note that there could be several exits. Each exit is then represented by an empty room that is connected to the Exit Node.

Victory Node This node can be defined as $I_1 = \{|P| = 0, win(), false\}$. The set P being the set of players (see section), whenever this set is empty (i.e. there are no more players in the game), the game is won. Once the game is won, there is no game anymore and the *Repeat* value doesn't mean much. This node doesn't require to be connected to any node. However, in order to respect the fact that the Escape Game Graph is connected, we will connect it to the Exit node.

The Victory node may seem useless. Indeed, with the Exit node removing players, the game could calculate by itself whether all players are gone. However, in some situation the goal is different. The objective of an Escape Game is **usually** to exit, but not always. With this Victory node, it is possible to alter the Victory condition without altering the rest of the game mechanics.

3.3.8 Incidence function

The incidence function ϕ is the function that maps every edge $w_x \in W$ to an ordered pair of nodes (from the sets S, C, R, L, Z and I).

There are, however, rules to respect as one node cannot be linked to any other. The table 3.2 lists all the allowed links.

Source Node	Destination Node	Set space	meaning
Skill	Role	$\{W \longrightarrow S \times L\}$	The role requires the skill
Clue	Role	$\{W \longrightarrow C \times Z\}$	The role requires the clues
Role	Room	$\{W \longrightarrow L \times R\}$	The room is a starting position
Room	Role	$\{W \longrightarrow R \times L\}$	The role is located in the room
Role	Puzzle	$\{W \longrightarrow L \times Z\}$	The puzzle requires the role
Puzzle	Puzzle	$\{W \longrightarrow Z \times Z\}$	The puzzle reward is another puzzle
Puzzle	Clue	$\{W \longrightarrow Z \times C\}$	The puzzle reward is a clue
Puzzle	Room	$\{W \longrightarrow Z \times R\}$	The puzzle reward is a door opening
Room	Clue	$\{W \longrightarrow R \times C\}$	The clue is in the room
Room	Puzzle	$\{W \longrightarrow R \times Z\}$	The puzzle is in the room
Room	Room	$\{W \longrightarrow R \times R\}$	The room links to another room
Meta	Any	$\{W \longrightarrow I \times \{S, C, R, L, Z, I\}\}$	Depends on the condition and action

Table 3.2: Allowed links for incidence function

3.3.9 Example

The following example illustrates how those rules can be used in order to generate an Escape Game Graph.

Let us build an Escape Game, containing 2 rooms. Players start in the first room and must solve the last puzzle in order to open the exit door. This last puzzle is a lock pair on the exit door, therefore requiring 2 keys in order to be opened. Both keys must be turned together, requiring 2 players to coordinate.

The first key is to be found after opening a chest, located in the first room. The chest is locked by a three digit lock, thus requiring 4 information clues (3 digits and the order in which to place them). Each digit is painted on a wall in the second room, in a different color. Next to the chest are three painted circles, matching the digit colors. The circles represent the order in which to input the digits into the lock.

For the second key, it is the reward of another puzzle, located in the second room. The key is at the bottom of a water tank. Players have to figure out how to get the key. They will find a magnet in the first room and a string attached to a stick in the second room, next to the water tank. Players must understand that they need to attach the magnet to the string, allowing them to "fish" the key.

Using the Escape Game Graph definition earlier provided, we can formalize this game description as follow :

- Rooms are R_1, R_2 and R_3. R_1 is the starting room. R_3 is the exit room.
- L_0, the standard starting position node, is linked to R_1.
- The chest is puzzle Z_1 :
  - It requires one player to be solved, therefore role $L_{1.1}$ is linked to the puzzle.
  - Digit painted clues are clues $C_{1.1}$, $C_{1.2}$ and $C1.3$ and are located in R_2.
  - Painted circles are clue $C_{1.4}$, located in R_1.
  - All those clues are linked to $L_{1.1}$.
  - Required skills, connected to $L_{1.1}$ are :
    S_1 Observation/Searching, to find the clues.
    S_2 Deduction, to figure out what to do with them .
    S_3 Object Movement/Alignment, to manipulate the digit-lock on the chest.
  - The reward for Z_1 is a key, clue $C_{3.1}$ to the final puzzle Z_3.
- The water tank is puzzle Z_2 :
  - It requires one player to be solved, therefore role $L_{2.1}$ is linked to the puzzle.
  - The magnet is clue $C_{2.1}$ and located in room R_1.
  - The string and stick are clue $C_{2.2}$ and located in room R_2.
  - Those clues are linked to $L_{2.1}$.
  - Required skills, connected to $L_{2.1}$, are :
    S_1 Observation/Searching, to find the clues.
    S_4 Combination, to build the "fishing rod".
    S_5 Self-Motion/Agility, to "fish" the key.
  - The reward for Z_2 is a key, clue $C_{3.2}$ to the final puzzle Z_3.
- The Locks on the exit door are puzzle Z_3 :
  - It requires one player in front of each lock, therefore requiring 2 roles : $L_{3.1}$ and $L_{3.2}$.
  - The first key $Z_{3.1}$ is linked to role $L_{3.1}$
  - The second key $Z_{3.2}$ is linked to role $L_{3.2}$
  - Required skills, connected to both roles, are :
    S_3 Object Movement/Alignment, to turn the keys
    S_6 Distributed Knowledge/Information Sharing, to coordinate on when to do it.
  - R_3 is the Exit room, and the reward of puzzle Z_3.

Now that the objects present in the Game have been defined, we can define our Escape Game E as the graph :

$$
\begin{aligned}
E &= (S, C, R, L, Z, I, W\phi) \\
S &= (S_1, S_2, S_3, S_5, S_6) \\
C &= (C_{1.1}, C_{1.2}, C_{1.3}, C_{1.4}, C_{2.1}, C_{2.2}, C_{3.1}, C_{3.2}) \\
R &= (R_1, R_2, R_3) \\
L &= (L_{1.1}, L_{2.1}, L_{3.1}, L_{3.2}) \\
Z &= (Z_1, Z_2, Z_3) \\
I &= (I_0, I_1) \\
W &= (w_1, w_2, w_3, ..., w_{31}, w_{32})
\end{aligned}
$$

Figure 3.1 illustrates the Escape Game Graph representing our game. The dashed rectangles correspond to room boundaries. For the sake of clarity, we decided to omit the skill nodes and the edges linking them. Adding them into the graph would only make it harder to read. The next graphical representations will usually leave skills out of the way.

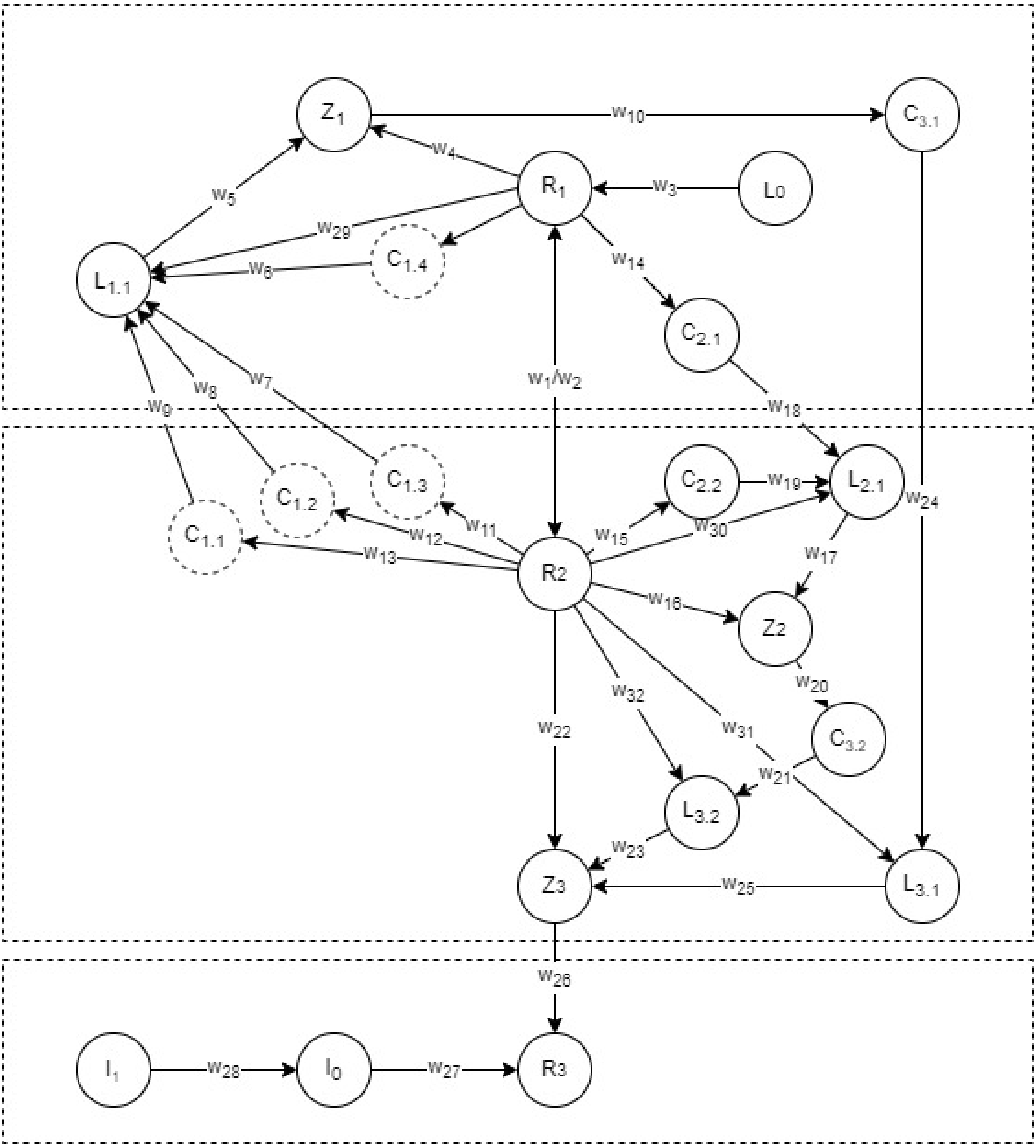


Figure 3.1: Escape Game Graph Example

3.4 The Game Session Frame Graph

3.4.1 Game Sessions

The Escape Game Graph previously introduced allows us to describe an Escape Game from a static perspective. It can be considered as the blueprint of an Escape Game. While this graph by itself already presents many relevant information, it would be interesting to be able to model a game while being played.

In order to present a complete model, we will now introduce the concept of Game Session (GS) and Game Session Frame (GSF). A Game Session is a mathematical representation of an Escape Game when it is played by real players, from start to finish. A Game Session Frame is a mathematical representation of an in-progress Escape Game, at a given moment. It can be considered as snapshot of the game in a given point in time.

A Game Session is therefore a set of Game Session Frames, linked together by the decisions players make in order to progress in the game.

In this section, we will first describe the Game Session Frame components. We will then provide ways to calculate how the possible actions[12] can be found from a Game Session Frame and the Escape Game Graph. For each of these actions, we shall define the rules for generating a new Game Session Frame from the previous one, the selected action and the Escape Game Graph. Finally, we will describe how a Game Session tree can be build and present the Game Session forest for a given Escape Game Graph.

3.4.2 Game Session Frame

A Game Session Frame F_k is a mixed, simple graph[13]. It contains two edge types (oriented and non-oriented), 5 node types (Rooms, Players, Skills, Clues and Puzzles) and 2 incidence functions (map and objects).

It can be represented as

$$F_k = (E_m, E_f, R_f, P_f, S_f, C_f, Z_f, \phi_m, \phi_o)$$

where :

- E_m is the oriented edge set, used for the map,
- E_f is the non-oriented edge set,
- R_f is the discovered room set,
- P_f is the active player set,
- S_f is the player's skill set,
- C_f is the discovered but unused clue set,
- Z_f is the discovered but unsolved puzzle set,
- ϕ_m is the incidence function representing the map,
- ϕ_o is the incidence function representing the objects.

3.4.3 Oriented edges

Oriented edges will be used in order to represent available player path. They will connect rooms together and indicate whether a player can move to another room or not. These edges will be noted E_{mx} (where x is the oriented edge number) and belong to the set E_m.

If players can move from one room to another and come back, then two edges will exist, because they are oriented. In order to ease the graphical representation, this situation may be presented as a double arrow.

[12] The actions have been defined in section 3.1.2 Actions

[13] while usually connected, it is possible that some event would close a door and that players loose access to a room, disconnecting the graph. It is also possible that players are separated from each other in different rooms that are not reachable (yet, or any more).

3.4.4 Non-oriented edges

Non-oriented edges will be used to represent all the other connections between nodes, as orientation does not matter. Saying that a puzzle is in a room or that a room contains a puzzle makes no difference in terms of semantics.

They may represent :

- The location of an item (edge between an item (player, clue or puzzle) and a room),
- A player possessing a skill (edge between a player and a skill),
- A player having knowledge of an information clue (edge between a player and a clue),
- A player carrying a clue (edge between a player and a clue),
- A player carrying a puzzle (edge between a player and a puzzle).

They will be noted E_{fx} (where x is the non-oriented edge number) and belong to the set E_f.

3.4.5 Rooms

Rooms are the physical space that players evolve in. Pretty much any other object is connected to a room to represent the fact that it is located somewhere. Rooms are connected through each other by oriented edges to represent the possible player path.

We will use the same objects as in the Escape Game Graph in which rooms are noted R_x and $R_x \in R$.

When a room is present in a Game Session Frame, it means players have discovered it. The Escape Game may contain rooms that have not been discovered yet. Therefore the Game Session Frame room set R_f is always a subset of the Escape Game Graph room set R.

3.4.6 Players

Players are the human beings that will play the Escape Game. They will evolve through the game with the objective of exiting it. They will initiate actions the system will respond to in order to advance in their progression. As stated earlier, the Game is usually won when all players have exited the game. For other win conditions, see section 3.3.7 Victory Node.

Players will be noted P_x where x is the player number. They belong to the set P_f. This set will reduce every time a player manages to exit the game.

Players are always located in a room, therefore there will always be one (and only-one) non-oriented edge from each player to a room.

Each player possess a unique skill set, so there will be non-oriented edges from each player to the skills they possess.

Players can gather knowledge from information clues, this will be represented as non-oriented edges going from the players to the clues they are aware of. They can also carry item clues. So a non-oriented edge will connect a player to each clue he/she is carrying.

3.4.7 Skills

Skills in the Game Session Frame correspond to the skills players possess. Similar to the Skills in the Escape Game Graph, they all belong to the Full Skill Set S^F. Skills in the Game Session Frame are noted S_{fx} where x is the skill number and $S_{fx} \in S_f \subseteq S^F$.

Skills are linked to players with non-oriented edges depending on their competences. They will be used by the system to validate whether a given player can solve a puzzle.

From the skill set S_f, it is already possible to check whether the game is impossible. If the skill set in the Escape Game Graph is not a subset of the Game Session Frame skill set, we already know that players do not have the skills to complete all the puzzle, therefore they will probably not be able to win the game. That is, if $S \nsubseteq S_f$, the game is impossible.

3.4.8 Clues

Clues, as stated before, are used by players to solve puzzles. They can be information or items. Any clue present in a Game Session Frame represent a clue that has been discovered but not used yet. Clues will therefore appear in a Game Session Frame, then disappear when the puzzle they relate to is solved.

Since they represent the same objects as in the Escape Game Graph, clues in the Game Session Frame Graph will be noted the same way, $C_{x.y}$. All clues present in the Game Session Frame Graph will be in the set C_f and $C_f \subseteq C$.

Clues may be located in a room or carried by a player. Informative clues are known by players. Item clues therefore only possess one non-oriented edge that goes either to a Room or to a Player. Informative clues possess one non-oriented edge to the room they belong to, and one non-oriented edge to each player that knows about the clue.

3.4.9 Puzzles

Puzzles are to be solved by the players using clues and skills. Whenever a player makes an attempt to solve a puzzle, the current Game Session Frame Graph and the Escape Game Graph will be used to check whether the puzzle can be solved. See section 3.5.2 Attempt for details on how these checks can be performed.

Puzzles nodes in a Game Session Frame Graph correspond to puzzles that have been found but not solved yet. Whenever a puzzle is solved by the players, it will be removed from the Game Session Frame Graph. Puzzles are linked to at least one room[14].

Again, they are the same items as the puzzles in the Escape Game Graph. Therefore, any puzzle in the Game Session Frame Graph is $Z_x \in Z_f \subseteq Z$.

3.4.10 Map incidence function

The oriented incidence function ϕ_m is the function that maps every edge from the set E_m to an ordered pair of nodes from the Room set R_f. It represent the available path for the players in the current game frame. In order to ease calculation whenever a path is required to be found, it is possible to produce a reduced graph from any given Game Session Frame Graph. The Map function provides this functionality and is defined as follow :

Function 3.4.1 (Map).

$$Map(F_k) = (R_f, E_m, \phi_m)$$

This is a valid oriented simple graph containing nodes (R_f), edges (E_o) and an incidence function (ϕ_m) that maps the edges to an oriented pair of nodes. The Map function simply extracts the required items from a given Game Session Frame Graph. Mainly, it can be used in order to check whether a path exists from a given Room to another one.

3.4.11 Object Incidence Function

The Object Incidence Function ϕ_o is the function that maps every edge from the set E_f to a pair of nodes from the sets $R_f, P_f, S_f C_f$, and Z_f. There are, however, rules to respect as one node cannot be linked to any other. The table 3.3 lists all the allowed links. Note that source and destination nodes can be switched as the edges are non-oriented.

[14]It is possible that a puzzle is located in two rooms, therefore requiring multiple players to solve it

Source Node	Destination Node	Set space	description
Room	Player	$\{R_f \times P_f\}$	The player is in the room
Room	Clue	$\{R_f \times C_f\}$	The clue is in the room
Room	Puzzle	$\{R_f \times Z_f\}$	The puzzle is in the room
Player	Skill	$\{P_f \times S_f\}$	The player possesses the skill
Player	Clue	$\{P_f \times C_f\}$	The player carries (or memorized) the clue

Table 3.3: Allowed links for the Object Incidence Function

3.4.12 Example

To illustrate the Game Session Frame Graph, let us remind about the Escape Game Graph example, illustrated by Figure 3.1 Escape Game Graph Example. We defined an Escape Game and drew its graph.

We will now illustrate the Game during a game session. At some point, 2 players started the game, P_1 and P_2. Both have all the required skills for all the puzzles. They explored around and solved the chest puzzle (Z_1), P_1 gathered the reward (a key for the final puzzle). P_2 wandered around and found the string attached to a stick, he picked it up. They saw the magnet but left it in the first room, having not yet discovered its usage. All the puzzle have been found.

The Game Session Frame Graph is actually much simpler than the Escape Game Graph :

$$
\begin{aligned}
F_k &= (E_m, E_f, R_f, P_f, S_f, C_f, Z_f, \phi_m, \phi_o) \\
E_m &= (E_{m1}, E_{m2}) \\
E_f &= (E_{f1}, E_{f2}, E_{f3}, E_{f4}, E_{f5}, E_{f6}, E_{f7}) \\
R_f &= (R_1, R_2) \\
P_f &= (P_1, P_2) \\
S_f &= (S_1, S_2, S_3, S_4, S_5, S_6) \\
C_f &= (C_{2.1}, C_{2.2}, C_{3.1}) \\
Z_f &= (Z_2, Z_3) \\
\phi_m(E_{m1}) &= (R_1, R_2), \phi(E_{m2}) = (R_2, R_1) \\
\phi_o(E_{f1}) &= (R_1, C_{2.1}), \phi_o(E_{f2}) = (R_1, P_1), \phi_o(E_{f3}) = (P_1, C_{3.1}), \phi_o(E_{f4}) = (P_2, C_{2.2}), \\
\phi_o(E_{f5}) &= (R_2, P_2), \phi_o(E_{f6}) = (R_2, Z_2), \phi_o(E_{f7}) = (R_2, Z_3)
\end{aligned}
$$

Figure 3.2 illustrates the graph. This is the current game situation and is actually pretty easy to read. Again, the skills have not been represented.

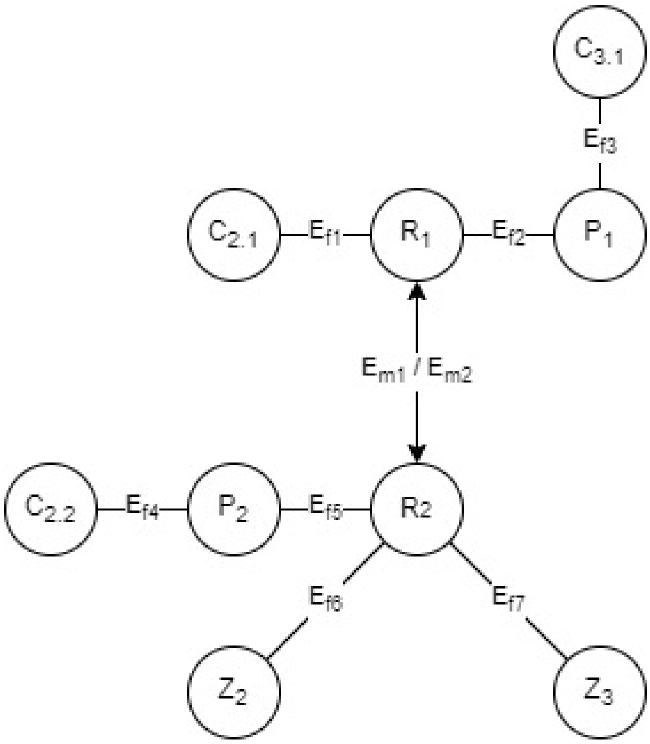


Figure 3.2: Game Session Graph Example

3.5 Actions and reactions

The Game Session Frame Graph has been introduced in the previous section, providing the possibility to represent an Escape Game at a given moment of its execution by players. We shall now define how to calculate the possible actions for the players, based on a Game Session Frame Graph and the Escape Game Graph. Those actions have been defined in section 3.1.2 Actions. For each action, we will determine how to calculate whether it is a possible action for the current frame.

3.5.1 The action set

For any given Game Session Frame Graph F_k, there is a set of possible actions $A(F_k)$ that can be calculated. The set contains all the actions players can make at this particular moment of the game. Each action will be noted A_x, where x is the action number and $A_x \in A(F_k)$. Any action A_x is an ordered set containing different item types. The type and ordering of the items are unique to each action type.

Each action type is a subset of the action set $A(F_k)$. Therefore the action set is the union of all action type sets.

The Find function

Before calculating possible actions, we must introduce the Find function. As an Escape Game is a graph, we already know that edges are linked through nodes using the graph incidence function. The incidence function is a mapping that links one edge to a pair (ordered or not) of nodes. In this work, we will introduce a new function, based on the incidence function. The function $Find()$ is a function that maps any item (node or edge) or set to a set containing all the mappings in which this item is present in a given incidence function.

The $Find$ function can be defined as :

Function 3.5.1 (Find).

$$Find(G, \phi, I)$$

where

- G is the Graph containing the items to find.
- ϕ is the incidence function to use from the graph.

- I is an item (node, edge or set) existing in the graph.

As an example, let us define the graph G as :

$$\begin{aligned} G &= (V, W, E, \phi) \\ V(G) &= \{v1, v2, v3, v4\} \\ W(G) &= \{w1, w2\} \\ E(G) &= \{e1, e2, e3, e4, e5\} \\ \phi_G(e1) &= (v1, v2), \phi_G(e2) = (v1, v3), \phi_G(e3) = (v2, v4), \\ \phi_G(e4) &= (v3, w1), \phi_G(e5) = (w1, w2) \end{aligned}$$

where

- V and W are sets containing nodes of different types.
- E is the edge set.
- ϕ is the incidence function.

We define the $Find()$ function as follow :

Function	Resulting set
$Find(G, \phi_G, e1)$	$\{e1, v1, v2\}$
$Find(G, \phi_G, e2)$	$\{e2, v1, v3\}$
$Find(G, \phi_G, e3)$	$\{e3, v2, v4\}$
$Find(G, \phi_G, e4)$	$\{e4, v3, w1\}$
$Find(G, \phi_G, e5)$	$\{e5, w1, w2\}$
$Find(G, \phi_G, v1)$	$\{e1, e2, v1, v2, v3\}$
$Find(G, \phi_G, v2)$	$\{e1, e3, v1, v2, v4\}$
$Find(G, \phi_G, v3)$	$\{e2, v1, v3\}$
$Find(G, \phi_G, v4)$	$\{e3, v2, v4\}$
$Find(G, \phi_G, w1)$	$\{e3, v3, w1\}$
$Find(G, \phi_G, w2)$	$\{e5, w1, w2\}$
$Find(G, \phi_G, V)$	$\{e1, e2, e3, e4, v1, v2, v3, v4, w1\}$
$Find(G, \phi_G, W)$	$\{e4, e5, v3, w1, w2\}$
$Find(G, \phi_G, E)$	$\{e1, e2, e3, e4, e5, v1, v2, v3, v4, w1, w2\}$

Table 3.4: The Find function

In other words, the $Find()$ function allows us to get all objects that are connected to the ones passed as parameter.

It is also possible to pass custom subsets of the graph to the $Find$ function. It will then return a union of the items :

$$\begin{gathered} Find(G, \phi_G, \{e1, e3, e5\}) \\ = \\ Find(G, \phi_G, e1) \cup Find(G, \phi_G, e3) \cup Find(G, \phi_G, e5) \\ = \\ \{e1, e3, e4, v1, v2, v4, w1, w2\} \end{gathered}$$

If the graph is oriented, the $Find$ function behaviour is the same.

Enter

This action can be executed only once by the players, at the game startup. Therefore this action will never be present in any action set, except the first one. The *Enter* action is always the first possible action in the game and can be defined as :

$$A_0 = \{P\}$$

where P is the player set.

Exit Game

As stated in section 3.3.7 Exit Node, players leave the game when in a specific room. Therefore leaving the game is not a specific action but only a movement to this specific room.

Move

This action allows players to move from one room to another. The set $A(F_k)^M$ set contains all movement actions. A move action A_x is defined as :

$$A_x = \{P_y, R_z\}$$

The ordered set is a pair containing a player P_y as first item then a room R_z as second item, indicating that the player can move from its current location to the provided room.

In order to find all the possible moves a player can make, we first need to find the player's current room. For this purpose, we define the function $Location(P_y)$, which is a shorter way for :

Function 3.5.2 (Location)**.**

$$Location(P_y) = Find(F_k, \phi_o, P_y) \cap R$$

The $Find$ function will return a set containing all objects directly connected to the player. Since a player can be directly connected to only one room at any given moment, intersecting this set with the set of all rooms results in a set containing only the room the player is connected to. Therefore, $Location(P_y)$ is a room, precisely the room in which the player is located.

Once the player location has been found, it is then possible to use a graph traversal algorithm on the $Map(F_k)$ graph (which is a graph containing only rooms and path linking them) in order to find all rooms that are reachable from $Location(P_y)$. Any Room node R_z that is found by the algorithm will generate an action $A_x = \{P_y, R_z\}$ that is to be included in the set of possible move actions $A(F_k)^M$.

The choice of the algorithm is not relevant. Any algorithm ensuring all reachable room nodes from R_z will be reached is sufficient to generate the correct action set. The path from the player's location to any room he/she can move to is irrelevant as well. Knowing the player can move to the given room is sufficient to add the corresponding action to the move action set. Assuming a correct graph traversal algorithm has been chosen, we can define the function $Reachable()$ as the function that returns a reachable room set from a given room and a map. It is defined as :

Function 3.5.3 (Reachable)**.**

$$Reachable(r, M)$$

where :

- r is a room node existing in M.
- M is a graph such a the one defined by the $Map()$ function.
- The resulting set contains all reachable rooms in M from r, including the room r.

Of course, this search has to be done for each player in the game. Therefore, we can define the possible move action set $A(F_k)^M$ as follow :

$$\begin{aligned}
&A(F_k)^M = \\
&\quad A_x | \forall p \in P, \\
&\qquad \forall r \in Reachable(Location(p), Map(F_k)), \\
&\qquad A_x = \{p, r\}
\end{aligned}$$

Search

This action will allow players to discover the items that exist in the game but are still hidden to them. The set $A(F_k)^S$ contains all search actions. A Search Action is defined as :

$$A_x = \{P_y\}$$

Because a player can only make a search in the room he/she is located, finding the player's current room is the only required step for the action to be generated. This is done by reusing the $Location(P_y)$ function defined earlier.

Each player can perform a search, therefore the possible search action set $A(F_k)^S$ will contain one search action for each player still in the game :

$$\begin{aligned}
&A(F_k)^S = \\
&\quad A_x | \forall p \in P, \\
&\qquad A_x = \{p\}
\end{aligned}$$

Attempt

This action becomes available when a player or a group of player have the possibility to solve a puzzle. Since we only care about generating the possible actions, we do not know (yet) if the players are in conditions to solve it. The attempt action set is $A(F_k)^A$ and an attempt action is :

$$A_x = \{Z_y, P(Z_y)\}$$

where :

- Z_y is the puzzle to solve
- $P(Z_y)$ is the set of players that can and want to try solving the puzzle

Before building all the possible attempt actions, we need to create a function that maps every reachable puzzle Z_x to the set of players that may try to solve it :

$$Attempts \equiv Z_x \longrightarrow P(Z_x)$$

A player P_y can make an attempt on a puzzle Z_x if he/she is in the same room R_z as a role $L_{x.a}$ defined for the puzzle (linked to the puzzle in the Escape Game Graph). Even if the player doesn't have the required skills or clues to solve it, it is possible to try.

First, we need to find the player's current room, using the $Location(P_y)$ function. Then, we find all the roles connected to this room in the Escape Game Graph E :

$$Roles(P_y) = Find(E, \phi, Location(P_y)) \cap L$$

$Find(E, \phi, Location(P_y))$ will return a set containing all objects connected to the current player's room in the Escape Game Graph. Intersecting this set with the Role set L will return a set containing all roles in the player's room.

Finally, we can find the puzzles linked to those roles :

$$Puzzles(P_y) = Find(E, \phi, Roles(P_y)) \cap Z$$

Using the same approach, we find all objects connected to the roles, then the intersection with the puzzle set Z will return the reachable puzzle set. Each puzzle Z_x in $Puzzles(P_y)$ will be added to the mapping function *Attempts*. Once done for every player and every puzzle, it remains to build all the possible combinations for players grouping together.

For example, if players P_1 and P_2 have the possibility to solve puzzle Z_1, the mapping function would be generated as follow :

$$Attempts = \{Z_1 \longrightarrow \{P_1, P_2\}\}$$

Players actually have the following choices :

- Player P_1 makes the attempt
- Player P_2 makes the attempt
- P_1 and P_2 make the attempt together

Therefore, we need to generate the following actions :

- $A_1 = Attempt(Z_1, \{P_1\})$
- $A_2 = Attempt(Z_1, \{P_2\})$
- $A_3 = Attempt(Z_1, \{P_1, P_2\})$

The different possibilities correspond to the power set[15] $\mathcal{P}(P(Z_y))$ (excluding the empty set) generated from the mapping set in *Attempts*.

Therefore, the attempt action set can be expressed as :

$$\begin{aligned} A(F_k)^S = \\ & A_x | \forall (Z_y \longrightarrow P(Z_y)) \in Attempts, \\ & \quad \forall ps \in \mathcal{P}(P(Z_y)) \setminus \emptyset, \\ & \quad A_x = \{Z_y, ps\} \end{aligned}$$

Inventory

At any moment during the game, players can perform action on their inventory. They can pick up or drop a clue. They can also memorize an information clue.

The inventory action set $A(F_k)^I$ contains actions A_x such that :

$$A_x = \{P_y, C_{z.a}\}$$

If the player is already carrying the clue, then the action is a request to drop it in the room where the player is located. If the player isn't carrying the clue, then it is a request to take it. Information clues can't be dropped, so it is always a request to memorize it.

[15] The set containing all subsets

Clue subsets Since clues will be separated and filtered according to their types, we can define the subsets C^{item} and C^{info}, respectively the set containing only the item clues and the set containing only the information clues. Both are a subset of C and they are defined as :

$$
\begin{aligned}
C^{item} &\subseteq C \\
C^{info} &\subseteq C \\
C^{item} \cup C^{info} &= C \\
C^{item} \cap C^{info} &= \emptyset \\
C^{item} &= (C_{x.y} \in C | C_{x.y}.ClueType = Item) \\
C^{info} &= (C_{x.y} \in C | C_{x.y}.ClueType = Info)
\end{aligned}
$$

To generate those actions, there are three rules :

R.1 Any item clue the player is carrying can be dropped

R.2 Any item clue in the same room as the player can be picked-up

R.3 Any information clue the player isn't aware of yet and in the same room as the player can be memorized.

Each one of these rules will generate a specific action set, itself a subset of the Inventory action set. We can then redefine the Inventory action set as :

$$A(F_k)^I = A(F_k)^{ID} \cup A(F_k)^{IP} \cup A(F_k)^{IM}$$

Each of these sets is built according to the predefined rules.

Rule R.1, Drop The clue set that can be dropped is actually the clue set the player is already carrying :

$$Find(F_k, \phi_o, P_y) \cap C^{item}$$

Intersecting player's connected items with the item clue subset will result in a set containing item clues connected to the player.

Any clue c in this set will generate a drop action A_x and we can define the set $A(F_k)^{ID}$ as :

$$
\begin{aligned}
&A(F_k)^{ID} = \\
&\qquad A_x | \forall c \in (Find(F_k, \phi_o, P_y) \cap C^{item}), \\
&\qquad\quad A_x = \{P_y, c\}
\end{aligned}
$$

Rule R.2, Pick-up The clue set that can be picked-up is the set of items in the same room as the player, excluding the information clue. We will reuse the $Location(P_y)$ function defined earlier :

$$Find(F_k, \phi_o, Location(P_y) \cap C^{item}$$

Any clue in this set will generate a pickup action A_x and the set $A(F_k)^{IP}$ can be defined as :

$$
\begin{aligned}
&A(F_k)^{IP} = \\
&\qquad A_x | \forall c \in Find(F_k, \phi_o, Location(P_y) \cap C^{item}, \\
&\qquad\quad A_x = \{P_y, c\}
\end{aligned}
$$

Rule R.3, Memorize

This is the set containing information clues located in the same room as the player and that the player hasn't already memorized :

$$(Find(F_k, \phi_o, Location(P_y) \cap C^{info})(Find(F_k, \phi_o, P_y) \cap C^{info})$$

Clues c in this set will generate an action A_x that allows to define the set $A(F_k)^{IM}$ as :

$$\begin{aligned} &A(F_k)^{IM} = \\ &\quad A_x | \forall c \in Find(F_k, \phi_o, Location(P_y) \cap C^{info}) \setminus (Find(F_k, \phi_o, P_y) \cap C^{info}, \\ &\quad A_x = \{P_y, c\} \end{aligned}$$

Communicate

Players have the ability to exchange and share information or item clues. To model this, the Communicate action set $A(F_k)^C$ contains actions A_x that can be defined as :

$$A_x = \{C_{x.y}, P_a, P_b\}$$

This action represent the fact that player P_a has the possibility to give the clue $C_{x.y}$ to player P_b. If the clue is an information, it represent that fact that player P_a can "teach" the clue $C_{x.y}$ to player P_b. To reflect the differences between items and information, we can split the set $A(F_k)^C$ in two subsets, $A(F_k)^{CT}$ and $A(F_k)^{CF}$, respectively the items that can be given and the information that can be explained (or taught).

In order to communicate, we will assume players must be in the same physical location. Therefore, we will first build a mapping function $Others$ that maps a player to a set containing the other players in the same room :

$$Others \equiv P_x \longrightarrow P^{others}$$

The set P^{others} can be build as follow :

$$Find(F_k, \phi_o, Location(P_x)) \cap (P \setminus P_x)$$

For each player P_x as the source (left side of the arrow) of a mapping in $Others$, we can build its item and info sets P_x^{item} and P_x^{info} :

$$\begin{aligned} P_x^{item} &= Find(F_k, \phi_o, P_x) \cap C^{item} \\ P_x^{info} &= Find(F_k, \phi_o, P_x) \cap C^{info} \end{aligned}$$

The item set P_x^{item} is the set containing item clues carried by the player. All items in this set can be given to any player in the $Others$ mapping for the player P_x. Therefore, the Communicate actions set for items $A(F_k)^{CT}$ can be expressed as :

$$\begin{aligned} &A(F_k)^{CT} = \\ &\quad A_x | \forall (P_y, P^{others}) \in Others, \\ &\quad\quad \forall c \in P_y^{item}, \\ &\quad\quad \forall p \in P^{others}, \\ &\quad\quad A_x = \{c, P_y, p\} \end{aligned}$$

Likewise, the info set P_x^{info} is the set containing information clues known by the player. The player can share his/her knowledge only to players not aware of those information yet. The Communicate action set for information $A(F_k)^{CF}$ is slightly modified to exclude the players already aware of the information to share :

$$
\begin{aligned}
&A(F_k)^{CF} = \\
&\quad A_x | \forall (P_y, P^{others}) \in Others, \\
&\quad\quad \forall c \in P_y^{info}, \\
&\quad\quad \forall p \in P^{others} | (c \notin p^{info}), \\
&\quad\quad A_x = \{c, P_y, p\}
\end{aligned}
$$

3.5.2 The reaction set

We have established the possible actions and ways to calculate them from a Game Session Frame Graph and the Escape Game Graph. So, for each Game Session Frame Graph, there is a set of possible actions $A(F_k)$ from which players can choose. Once players have chosen an action, we need to generate a new Game Session Frame Graph to reflect the action's effect on the game. Sometimes, we have to generate multiple Graphs because the selected action may induce several possible reactions. Those newly generated graphs will be part of a set, the Reaction set, noted $R(A_x)$. Each action from the Action set $A(F_k)$ will generate a specific reaction set, containing one or more items. The items contained in the Reaction set are new Game Session Frame Graphs, for which we will be able to generate new action sets.

This section describes, for each Game Session Frame Graph F_k, how to generate the next graph F_{k+1} (or the next graph set) in reaction to a player's action. To simplify the model's mechanics, we need to enforce a constraint : a player or a player set only pick one action, players not involved in the action do nothing in the meantime. In the real world players will perform actions simultaneously. This is discussed in section 4.2.3 Player multithreading.

Generating a new graph

In order to generate new graphs from the previous situations, we need to introduce 3 functions :

1. Add
2. Remove
3. Update

Function : Add Whenever an item is to be added to game. The *Add* function will return a new graph based on the provided graph and the nodes and mapping to add.

Function 3.5.4 (ADD).

$$
\begin{gathered}
Add(G, N, M) \\
or \\
Add(G, M)
\end{gathered}
$$

where

- G is the Game Session Game Graph F_k in which to add N and M.
- N is a node set that exist in the Escape Game Graph E but not in G yet.
- M is a node pair set containing links to be added. All nodes in the mappings must exist in G or be part of N.

The function returns a new Game Session Frame Graph F_{k+1}, a copy of the graph F_k, in which the nodes N have been added. Edges of the proper type are added as well and the mapping function is updated accordingly.

There are three node types that can be added in a Game Session Graph :

1. Rooms, the new edges will be created in the E_m set and the mapping ϕ_m will be updated as well.
2. Puzzles, the new edges will be created in the E_f set and the mapping ϕ_o will be updated.
3. Clues, the new edges will be created in the E_f set and the mapping ϕ_o will be updated.

The second version of the *Add* function, without the specified nodes, allows us to add edges corresponding to the provided mappings. If a provided mapping contains two nodes of type Room, the edge will be added into E_m and ϕ_m will be updated accordingly. Otherwise, an edge will be added into E_f and ϕ_o will be updated.

Function : Remove Whenever an item is no longer needed in the game. The *Remove* function will return a new graph based on the provided graph and the nodes to be removed.

Function 3.5.5 (Remove)**.**

$$Remove(G, I)$$

where

- G is the Game Session Frame Graph F_k from which to remove the items.
- I is a set containing items already present in G. If a node is removed, all edges linked to it will also be removed and the mapping function updated accordingly.

The function will return a new Game Session Frame Graph F_{K+1} in which the provided item set is no longer present.

Items that can be removed from the game are :

1. Oriented Edges, whenever a door from the Escape Game Graph closes, an oriented edge from E_m is to be removed.
2. Puzzles : when solved, the puzzle disappears.
3. Clues : when the related puzzle is solved, clues for this puzzle are of no interest anymore.
4. Players : When in the Exit Room, players are removed from the game.

Obviously, any mapping in the incidence function containing the items to remove will also be removed, in order to keep consistency.

In order to remember which items have already been discovered (see later), we want to remember some of the removed items. To do so, we will create a set named *Removed* that will remember items not present in the game anymore. Only nodes of type Puzzle, Clue or Player will be present in the *Removed* set. Every time the *Remove* function will be used (and if the item type to remove is of one of those stated), the *Removed* set will collect removed nodes.

Function : Update The *Update* function allows to update an incidence function without touching anything else. It returns a new graph with the updated incidence function. It can be seen as edge redirecting.

Function 3.5.6 (Update)**.**

$$Update(G, E, M)$$

where

- G is the Game Session Frame Graph F_k to take the incidence function from.
- E is an edge in the graph.
- M is a new unordered node pair that will redirect the edge to another node pair.

The nodes in the new mapping must already exist prior to calling this function. This function will return a new Game Session Frame Graph F_{k+1} where it's object incidence function ϕ_o has been updated to reflect the edge redirection.

The operation is pretty straightforward :

$$\phi_o(E) = M$$

Table 3.5 Update function's rules lists the allowed mappings for the *Update* function.

original mapping	new mapping	description
Room to Player	New Room to Player	Player's movement
Clue to Room	Clue to Player	Player's pickup action
Clue to Player	Clue to Room	Player's drop action

Table 3.5: Update function's rules

Enter

Entering the game is an action that is executed only once, at game startup. It is the first action players will execute. The *Enter* action has bee defined as :

$$A_0 = \{P\}$$

Therefore we need to place the players and "build" the rooms around them. As a reminder, roles L_x connected to a room act as possible starting positions. Role L_0 is the standard starting position when all other starting positions are occupied.

We need to assign each player a starting position, taking into account the required skills for each position. Figure 3.3a illustrates an Escape Game having 2 start position L_0 and L_1, L_1 requiring a specific skill. Figure 3.3b illustrates 3 players, 2 of them possessing the skill required to start at position L_1. First, we want to gather all starting positions except the standard one L_0 into the set *Start* :

$$Start = (Find(E, \phi_o, R) \cap L^S) \setminus L_0$$

Roles in L^S are the starting position. We then exclude the standard position L_0.

For each starting position s in *Start*, there is a skill set, representing the required skills to be able to start from there. We shall gather those skills in the Start Requirement set $SR(s)$. There is one requirement set for each starting position. Likewise, each player possesses a skill set, indicating what he/she is capable of. The player skill set $PS(p)$ represents the skill set of player p and can be calculated as :

$$PS(p) = Find(F_k, \phi_o, p) \cap S$$

In order to validate the starting positions, we need to find a match between each Start Requirement Sets and the Player Skill Sets. Since the players can have more skills than required (but no less), it is sufficient to say that if $SR(s) \subseteq PS(p)$, then player p is eligible for starting position s. If all starting position can be filled by at least a different player, then it is possible to start the game. Remaining players will be put in the standard starting position L_0.

To solve this problem, we can build a bipartite graph in which the nodes are the Start Requirement Sets on one side and the Player Skill Sets on the other. Nodes are linked if $SR(s) \subseteq PS(p)$ (if a player is eligible for a role). The goal then is to find a maximum coupling including all of the Start Requirements Sets. If no such coupling exists, then all the requirements cannot be met with the current player set and the game cannot start. If a maximum coupling including all the Start Requirement sets can be found, then it is possible to start the game. Figure 3.3c illustrate the bipartite graph between the Start Requirements and the Player Skills.

Since we have to provide all possible reactions to a given action. Not only is it needed to find a maximum coupling including all the Start Requirement sets, it is also required to find them all. Each maximum coupling $C_x^M \in C^M$ will generate a different starting situation. Once the couplings have been found, we need to add a coupling between each remaining player (that is not yet part of a coupling) and the standard starting position L_0. We then need to create the rooms and place the players in it. A coupling pair $c \in C_x^M$ is an ordered node pair containing a player p and a role l (starting situation) : $c = (p, l)$. We will therefore gather all the rooms linked to the starting positions, create them, and create one edge between each player and its starting room. The room set to create is

$$Rooms = Find(E, \phi, L^S) \cap R$$

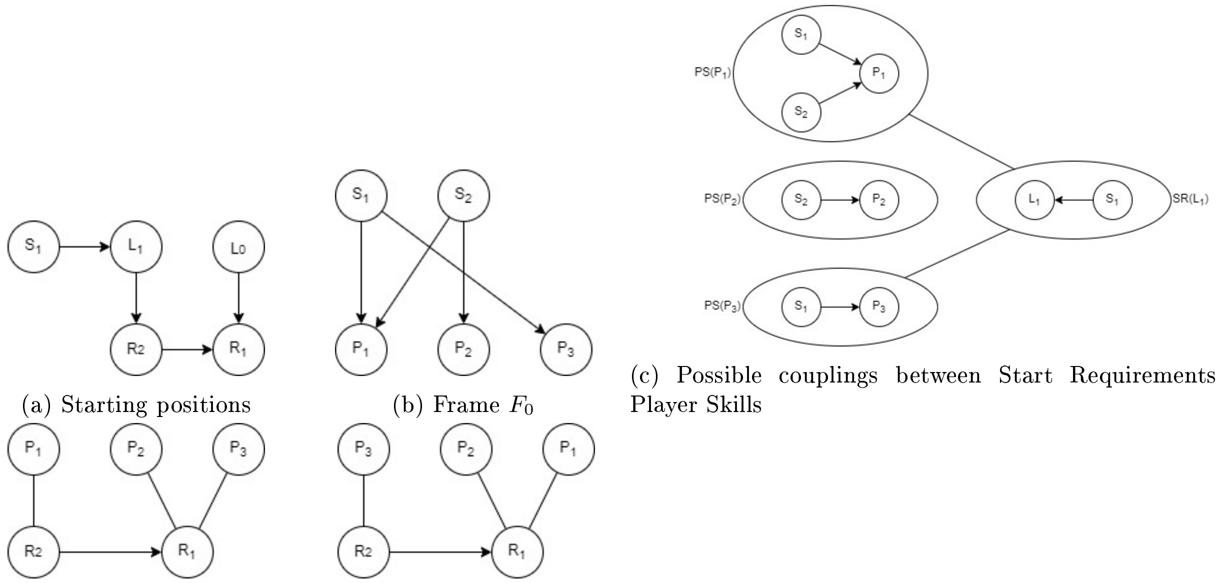
The mapping linking players to their starting room can be generated from the coupling C_x^M :

$$Mapping(C_x^M) = \forall (p, l) in C_x^M, \{p, Find(E, \phi, l) \cap R\}$$

The reaction set for entering the game can be expressed as :

$$R(A_x) = \forall C_x^M \in C^M, Add(F_k, Rooms, Mapping)$$

Figures 3.3d and 3.3e illustrates the 2 possible starting situation in the Reaction set.



(a) Starting positions

(b) Frame F_0

(c) Possible couplings between Start Requirements and Player Skills

(d) Starting situation 1

(e) Starting situation 2

Figure 3.3: Result of the Enter Action

Move

As defined earlier, a Movement action is an ordered set containing a player P_x and a reachable room R_y :

$$A_x = \{P_x, R_y\}$$

Because we already calculated it, we know the room is reachable (i.e. there is a valid path between the player and the room). We are not interested in which path the player is taking in order to move to that room, we are only interested in the action's result. The result is that the player is now located in room R_y and left the previous room he was in. In order to generate a new Game Session Graph reflecting the new game situation, we only need to redirect the edge going from the player to the previous room and make it go to the new room. This is done first by finding edge e, linking the player to its current room ($Location(P_y)$) :

$$e = Find(F_k, \phi_o, P_x) \cap Find(F_k, \phi_o, Location(P_x)) \cap E_f$$

The first intersect will return a set containing the player, the room and the edge linking them. Intersecting again with the non-oriented edge set will return only the edge. The *Update* function will then generate a new graph, reflecting the player's new location. It generates only one reaction in the reaction set :

$$R(A_x) = Update(F_k, e, \{P_x, R_y\})$$

Search

A search result can yield various results, depending on the player's location. Indeed, the point of performing a search action is for the players to discover new items (clues, puzzles or rooms) that haven't been found yet. However, the search will create only one new item in the game. So it is possible that a search action may generate several Game Session Frame Graph, depending on the object found.

The first thing to do is to build a set *Undiscovered* containing all the undiscovered items in the room players want to search. Then we generate one new Game Session Frame Graph for each item present in the *Undiscovered* set using the *Add* function.

As defined earlier, the *Search* action is defined as :

$$A_x = \{F_k, P_y\}$$

Only the following items can be discovered by a search :

1. Clues in the same room as the player.

2. Puzzles in the same room as the player.
3. Rooms connected to the player's current room.
4. Room connections that have not been found yet.

Available clues The available clues set Av_c, containing the clues to be discovered, can be found using the following calculation :

$$Av_c = c \in (Find(E, \phi, Location(P_y)) \cap C) | c \notin (C_f \cup Removed)$$

$Find(E, \phi, Location(P_y)) \cap C$ is a set containing all the clues linked to the player's room in the Escape Game Graph. We need then to filter out the clues that have already been discovered. $(C_f \cup Removed)$ is a set containing all the discovered clues, those still active and those that have been removed.

Available puzzles Processing the same way, the available puzzle set Av_z can be found using the following :

$$Av_z = z \in (Find(E, \phi, Location(P_y)) \cap Z) | z \notin (Z_f \cup Removed)$$

Again, we first find all the puzzles in the room, then remove those in the current Game Session Frame Graph as well as those already removed.

Connected rooms We first need to build the $UndiscoveredRooms$ set, containing all rooms possessing a path to or from the player's current location which have not yet been discovered :

$$UndiscoveredRooms = (P_y) = (Find(E, \phi, Location(P_y)) \cap R) R_f$$

Then we select the rooms accessible from $Location(P_y)$:

$$Av_r = r \in UndiscoveredRooms | \exists \phi(e) = \{Location(P_y), r\}$$

We find the connected rooms and remove those already existing in the current Game Session Frame Graph.

Room connections

Let us assume we have three rooms, all directly connected together. We would have the following in the Escape Game Graph :

$$\begin{aligned} R &= \{R_1, R_2, R_3, ...\} \\ W &= \{w_1, w_2, w_3, ...\} \\ \phi(w_1) = (R_1, R_2), \phi(w_2) &= (R_2, R_3), \phi(w_3) = (R_1, R_3), ... \end{aligned}$$

At some point in the game players search R_1 and find the next room R_2. Later on they search R_2 and find R_3. With the previously stated searches, the connection between R_1 and R_3 will never be found, because all rooms have been found. We need to look for undiscovered room connections.

We can first build the $DiscoveredRooms$ set, containing all rooms that have been discovered by the players (and linked to the player's current location) :

$$DiscoveredRooms(P_y) = Find(E, \phi, Location(P_y)) \cap R_f$$

Then, provided that there is a path between the rooms $r \in DiscoveredRooms(P_y)$ and the current player's room $Location(P_y)$, we check whether this path exists in the Game Session Frame Graph F_k :

$$Av_{rc} = r \in DiscoveredRooms(P_y) | \nexists \phi_m(e) = \{Location(P_y), r\}$$

The set Av_{rc} provides the missing room connections. First we select all the rooms connected to the current player's room in the Escape Game Graph that have already been discovered, then we keep only the rooms from which the current player's room doesn't provide a path.

Undiscovered items The *Undiscovered* set is then the union of the preceding sets :

$$Undiscovered = Av_c \cup Av_z \cup Av_r \cup Av_{rc}$$

Search results As previously stated, we will now generate a new Game Session Frame Graph F_{k+1} for each item u in *Undiscovered*. The creation rules are pretty straightforward :

Clues Clues are to be added directly in the room R_y :

$$F_{k+1} = Add(F_k, u, \{R_y, u\})$$

Puzzles Puzzles are to be added directly in the room R_y :

$$F_{k+1} = Add(F_k, u, \{R_y, u\})$$

Direct rooms Rooms from Av_r are to be added as unidirectional path. Players don't know in advance if there is way to come back to their current room :

$$F_{k+1} = Add(F_k, u, \{R_y, u\})$$

Room connections Rooms from Av_{rc} have already been discovered. What is left is to add an edge from R_y to those rooms :

$$F_{k+1} = Add(F_k, \{R_y, u\})$$

Attempt

An attempt reflects the fact that players want to try to solve a puzzle. It doesn't mean they will solve it. So the first thing to check is whether the attempt is successful or not. Depending on the result, the new graph will be generated.

It has been defined as :

$$A_x = \{Z_x, P(Z_x)\}$$

Attempt check In order to solve a puzzle, several conditions must be fulfilled :

C.1 There must be at least as many players as required by the puzzle.

C.2 The players attempting to solve the puzzle must be in the correct room.

C.3 The players attempting to solve the puzzle must possess the required skills.

C.4 The required clues for solving the puzzle must be in possession of the players attempting to solve it.

If any of these conditions is not met, the puzzle can't be solved and the attempt will result in a failure. Each of the previous conditions must be fulfilled in order to allow players to solve the puzzle. The puzzle itself doesn't contain any indications as what is required to solve it. We need to refer to the roles connected to Z_x. We will define the role set for the puzzle Z_x as :

$$L(Z_x) = Find(E, \phi, Zx) \cap L^P$$

checking C.1 We first need to find out how many roles are required to solve the puzzle :

$$N_R = |L(Z_x)|$$

then, if $|P(Z_x)| >= N_R$, the condition **C.1** is met.

checking C.2, C.3 and C.4 For each of the roles in $L(Z_x)$, we need to find a player possessing the same characteristics.

Namely, for each role, there must be a player that is :

- located in the proper room (**S.1**) and
- possess the required clues (**S.2**) and
- possess the required skills (**S.3**).

So, for each role $L_x \in L(Z_x)$, we can build the following sets :

S.1 $Find(E, \phi, L_x) \cap R$

S.2 $Find(E, \phi, L_x) \cap C$

S.3 $Find(E, \phi, L_x) \cap S$

The union of these sets is the Requirement Set for role L_x, noted $Rs(L_x)$. The Requirement Set for a Puzzle Z_x is $Rs(Z_x)$ and is a set containing all the roles Requirement Sets involved in the puzzle Z_x. There is one Requirement Set for each puzzle and these sets are constant throughout the game.

Then, for each player $p \in P(Z_x)$, we can build the following sets :

S.1 $Find(F_k, \phi_o, p) \cap R$

S.2 $Find(F_k, \phi_o, p) \cap C$

S.3 $Find(F_k, \phi_o, p) \cap S$

The union of these sets is the Player Capabilities Set for the player p, noted $Pc(p)$. Each player possesses a Player Capabilities Set, but it will change over time as they move from room to room and gather or drop clues.

In order to validate the puzzle conditions, the system needs to find a match between each Requirement Set for the puzzle and the Player Capabilities Sets. Since the players can carry more clues than required (but no less) and can have more skills than required (again, but no less), it is sufficient to say that if $Rs(L_x) \subseteq Pc(P_y)$, then player P_y is eligible for role L_x. If all roles for the puzzle are fulfilled by at least a different player, then the puzzle can be solved.

To solve this problem, we can build a bipartite graph in which the nodes are the Requirement Sets on one side and the Player Capabilities Sets on the other. Nodes are linked if $Rs(L_x) \subseteq Pc(P_x)$ (if a player is eligible for a role). The goal then is to find a maximum coupling including all of the Requirements Sets. If no such coupling exists, then all the requirements cannot be met with the current player set and the puzzle cannot be solved. If a maximum coupling including all the Requirement sets can be found, then the puzzle can be solved.

For example, let's imagine a big chest that has two locks on each of its sides. It requires two players to turn the keys in the locks at the same time. Let the chest be the puzzle Z_1 and the two keys be the clues $C_{1.1}$ and $C_{1.2}$. There would be also two roles $L_{1.1}$ and $L_{1.2}$, each one connected to one of the clues and the skill S_1 (corresponding to the ability to turn the key in the lock). If two players (P_1 and P_2) attempt to solve it together while P_1 owns the two keys, no coupling where all the Requirements are met exists. Players will first need to transfer one of the 2 keys to P_2 and attempt again. Then it will be possible to meet all the requirements and the puzzle can be solved.

The puzzle requirement set $Rs(Z_1)$ is :

$$
\begin{aligned}
&Rs(Z_x) = \\
&\qquad \{ \\
&\qquad Rs(L_1) = \{R_1, C_{1.1}, S_1\}, \\
&\qquad Rs(L_2) = \{R_1, C_{1.2}, S_1\} \\
&\qquad \}
\end{aligned}
$$

And the player's capability set $Pc(P_1)$ and $Pc(P_2)$ are, in the original situation :

$$
\begin{aligned}
Pc(P_1) &= \{R_1, C_{1.1}, C_{1.2}, S_1, S_2\} \\
Pc(P_2) &= \{R_1, C_{2.1}, S_1, S_3, S_4\}
\end{aligned}
$$

Both requirement sets are subsets of $Pc(P_1)$, but none of them is a subset of $Pc(P_2)$. Therefore, since P_1 can play only one role in the puzzle, the puzzle cannot be solved.

If P_1 was to give clue $C_{1.2}$ to P_2, the capability sets would then become :

$$Pc(P_1) = \{R_1, C_{1.1}, S_1, S_2\}$$
$$Pc(P_2) = \{R_1, C_{1.2}, C_{2.1}, S_1, S_3, S_4\}$$

In this new situation, $Rs(L_1) \subseteq Pc(P_1)$ and $Rs(L_2) \subseteq Pc(P_2)$, providing a maximum coupling including all the elements of $Rs(Z_x)$. The puzzle can therefore be solved.

Attempt failure Whenever the puzzle cannot be solved, the action made by the player is useless and the game situation didn't change. So, the newly generated Game Session Frame Graph is actually the same as the previous one :

$$F_{k+1} = F_k$$

Attempt success If the players manage to solve the puzzle, multiple actions must be taken in order to generate the next graph F_{k+1} :

1. All clues related to the puzzle are to be removed.
2. The puzzle must be removed.
3. The Reward must appear in the new graph

We will define 2 sets : *ToRemove* will contain items to remove from the current graph (clues and puzzle) and *ToAdd* will contain items to add and where to add them.

Removing clues The first action to take is to remove the clues linked to this puzzle. To do so, we can reuse the Puzzle Requirement Set $Rs(Z_x)$ as it already contains all the clues linked to the puzzle. We just need to keep only the clues and to put those in the *ToRemove* set :

$$ToRemove = Rs(Z_x) \cap C$$

Removing puzzle We add the puzzle in the *ToRemove* set. The set definition becomes :

$$ToRemove = (Rs(Z_x) \cap C) \cup Z_x$$

Granting reward As stated in the Puzzle node definition, the reward type can be a clue, a puzzle or a room. Also, there can be multiple rewards. The reward set can be defined as :

$$Rw = r \in \{R, C, Z\} | (\exists w | \phi(w) = \{Z_x, Re\})$$

We look for the items that have an incoming edge from the puzzle, they are the reward.

For each of those rewards, we need to add them to the new graph F_{k+1} :

$$F_{k+1} = Add(F_k, Rw, \{\forall Re \in Rw, \{R_y, Re\}\})$$

We add the reward set, providing a mapping from the puzzle's room to each reward.

Inventory

As a reminder, there are actually 3 possible inventory actions : drop, pick-up and memorize. For each of those, we need to define how the resulting graph will be calculated.

Drop The drop action was defined as :

$$A_x = \{P_y, c\}$$

The idea is that the player drops the clue in the room he/she is located. So all we have to do is find the edge linking the player and the clue, then redirect this edge to the room :

$$\begin{aligned} edge &= e \in E_f | \phi(e) = \{P_y, c\} \\ mapping &= \{c, Location(P_y)\} \\ F_{k+1} &= Update(F_k, edge, mapping) \end{aligned}$$

Pick-up The Pick-up action was defined as :

$$A_x = \{P_y, c\}$$

It is the same action as drop, but reversed. First we need to find the edge between the clue and the room, then redirect this edge :

$$\begin{aligned} edge &= e \in E_f | \phi(e) = \{Location(P_y), c\} \\ mapping &= \{c, P_y\} \\ F_{k+1} &= Update(F_k, edge, mapping) \end{aligned}$$

Memorize The Memorize action has been defined as :

$$A_x = \{P_y, c\}$$

The only thing to do here is adding a new edge between the player and the clue, because it will stay in the room :

$$F_{k+1} = Add(F_k, \{P_y, c\})$$

Communicate

Finally, the Communicate actions are to be calculated. As a reminder, the communication action set has been split in two sets, $A(F_k)^{CT}$ and $A(F_k)^{CF}$. For each of those, the rules are slightly different.

Exchange Actions in $A(F_k)^{CT}$ are defined as $A_x = \{c, P_y, p\}$, meaning that player P_y is about to give clue c to player p. To reflect this operation in the new graph, we need to modify the edge linking the clue c and P_y in order to make it go to p :

$$\begin{aligned} edge &= e \in E_f | \phi_o(e) = \{P_y, c\} \\ mapping &= \{c, p\} \\ F_{k+1} &= Update(F_k, edge, mapping) \end{aligned}$$

Communicate Actions in $A(F_k)^{CT}$ are defined as $A_x = \{c, P_y, p\}$, meaning that player P_y is about to reveal (or teach) clue c to player p. To reflect this operation in the new graph, we need to add an edge linking clue c and player p :

$$F_{k+1} = Add(F_k, \{c, p\})$$

3.6 The Game Session Forest

We now have the ability to calculate the possible action set $A(F_k)$ from a Game Session Frame Graph and to generate a reaction set $R(A_x)$ for each action. Thus providing us the ability to link Game Session Frame Graphs together. We can now introduce the Game Session Graph.

Game Session Graph

The Game Session Graph is a simple oriented and connected graph GS. It possess nodes that are all the possible Game Session Frame Graphs for a given Escape Game. The edges linking the nodes together are all the possible actions sets generated by all the Game Session Frame Graphs.

A Game Session Graph can be defined as :

$$GS = (F, R_A, A, M, \psi)$$

where:

- GS is the Game Session Graph.
- F is the Game Session Frame Graph set, containing all the game possibilities, each frame F_k is a node in the Game Session Graph.
- R_A is the Full Reaction set. Each Reaction set $R(A_x)$ is an element of R_A and is a node in the Game Session Graph.
- A is the Full Action set. Each Action set $A(F_k)$ is an edge in the Game Session Graph, linking an element of F to an element of R_A.
- M is the modification set. Each modification m is an edge in the Game Session Graph, linking an element of R_A to an element of F.
- ψ is the graph's incidence function.

The Game Session Graph will allow us to link several Game Session Frame graphs together. For each Game Session Frame Graph node F_k, there will be one leaving Action edge A_x from its action set $A(F_k)$. Those edges will go to a Reaction set node $R(A_x)$. Each possible reaction r in $R(A_x)$ generates a modification edge $m \in M$ to the next node $Fk + 1$.

The Game Session Graph is therefore a bi-colour tree, alternating Game Session Frame nodes and Reaction nodes. In this tree, the root node is the initialization node. This node is the starting situation, noted F_0 and is a Game Session Frame Graph containing only the players and their skills. The action set $A(F_0)$ contains one possible action A_0, *Enter*, meaning players want to start the game. The reaction set $R(A_0)$ contains all the possible starting situations. If there is only one starting position, the reaction set contains only one element. However if multiple starting situations are possible, the reaction set contains them all.

Because it is a tree, the Game Session Graph contains leaves. Leaves are the final game situation, offering a win to the players. The last Game Session Frame Graph node F_k is a graph in which the Victory Meta-Information node returns a *true* condition. We didn't define a loss condition, because we do not take time into account. It is therefore possible that the game never end. Indeed, players always have the possibility to search in any room, even if every item in it has been found. The Game Session Graph is an infinite tree.

3.6.1 The forest

We explained how the Game Session Graph is an infinite tree. Its root is one starting situation : A player set. Provided that the Game request a minimum and maximum amount of players, it is then possible to define all the possible starting situation.

If there is a maximum number of players and the skills are a finite set, then the numbers of way to start the game is a finite number. This number is large enough to understand that calculating every possible starting situation is not a realistic idea.

Because each possible combination of players and skills will generate its own Game Session Graph, we can say that the Full Game Session Graph is a forest, gathering all the possible Game Session Graphs.

4 Discussion

4.1 The Escape Game Framework

Framework construction

The attempt that has been made here is intended to provide a way to formalize an Escape Game. The main concept from Escape Games have been extracted from literature and personal experience. Concepts involving the game mechanics have been selected for formalization, while the others have been left behind. We certainly do not want to say that they are of lesser importance. An Escape Game without a theme or a narrative would work, but would also probably be boring to play.

Once the concept have been selected. Graphs have been chosen to represent a Game. They are actually well suited for this task, as they represented objects and how they are connected together. We had to define the connection rules. In order to present a basic framework, many things or ideas had to be abandoned. Those ideas are presented in section 4.2 Extensions.

Framework validation

The presented Framework has been validated only with trivial examples, which means it hasn't been validated, but seems to work. A proper validation would be required. Collecting some real-life Escape Games in order to check whether the Framework really can model a Game would be a first step in validating it.

Some known flaws already exist and the section 4.2 Extensions mentions some of them.

Usability

As such, the current framework is not of a real use. It is far too complex to be apprehended quickly and the generated Escape Game graphs are difficult to read. We introduced as example a very simple and short Escape Game but had to remove the skills and their connections for the sake of readability. However, we believe that the graphs and definitions can serve as a good basis for computer software. The first step of writing software is usually to formalize the problem, which is what we tried to do. The Game Session Frame graph seems to be the most readable, but has been tested only with small examples.

Also, there exist ways of simplifying the graphs in order to ease the deciphering. A Domain Specific Language could be produced that would simplify the visual representation of the Escape Game Graph. As an example, we could imagine that rooms aren't nodes anymore, but rectangles whose perimeter goes around all items inside them. This would eliminate all edges going from nodes to rooms. Any item in the rectangle would be considered linked to the node room, easing the visual without modifying the semantics.

Metrics

As it is, the Framework may already provide some interesting metrics. It is indeed possible to calculate how many players are required or the overall required skill set. It is possible to calculate how the skills are balanced in order to offer a varied experience to players. Those can be calculated directly from the graph and are therefore formally quantifiable, provided that the game has been modelled correctly. Some other metrics require calculations that have not been developed here. For example, deciding if the game is possible probably requires a dedicated algorithm.

The Game Session Forest could also provide metrics. If a specific game is monitored, then filling up the Game Session Graphs along the different real-life sessions would allow to see which path players usually take, which clues they gather first or which strategy they tend to use.

4.2 Extensions

The extensions presented here are ideas and/or concepts that have been abandoned during the Framework building. Either because it was of lesser interest of because it would have made the calculations much more complicated. We however wanted to discuss them for future works, should someone find an interest in any of them. They are presented in no particular order.

4.2.1 Skill level

Two players possessing the same skill do not necessarily master it the same way. Likewise, two puzzles requesting agility skills may not require the same skill level. An early abandoned extension to this framework would be to rate skills on a scale. While complicating the calculations to perform on puzzle attempts, it brings us closer to real life. Edges going from skills to roles could be labelled with a number, representing the required skill level. The same way, skills going to player nodes could also be labelled with a number, representing the player's skill level.

Educative Escape Games could benefit out of it, because the goal in such games is to increase a skill or extend the player's knowledge. Player skill level could therefore increase while they are progressing in the game, providing quantifiable metrics.

Of course, this would require to define a standard for quantifying skill levels, which is far beyond the scope of this work.

4.2.2 Time

Time is not involved in the pure game mechanics. However, an Escape Game in which you have an infinite amount of time might never finish while if you put a time constraint, you ensure that it will end at some point. It would possible to implement time in the framework, using the Game Session Graph. If each Action costs some time, then time can be a decreasing integer. Every Action edge traversal would decrease the time amount and when it reaches zero, the game is lost.

4.2.3 Player multithreading

When players must choose an action to perform, we decided that only one of them would perform the action while others would be waiting for its completion. Real life is thankfully not working like that. A player multi-threading extension would require to split the time even further, Game Session Frame Graphs would not represent an action result but a time frame. An action would take multiple time frames to be accomplished. Therefore a player starting an Action would be stuck for several frames in the action accomplishment. The others having the possibility to start other actions in the meantime. Having worked in software development for over 10 years, the author did not want to try making it work.

4.2.4 Non-Mandatory items

Sometimes, a puzzle possess non mandatory clues. If you take a padlock, requiring three digits in the proper order to be opened, you might think you need 4 clues (the 3 digits and the order). Reality is different, none of those clues are actually required. A padlock can be brute forced, like many other puzzles. Non-mandatory clues would be an extension exposing the brute force possibility. Only the edges going to the puzzle need to be modified to reflect a non-mandatory clue. The calculation of puzzle attempting and solving would be modified in order to reflect this.

5 Conclusion

The original idea behind this work was to check whether Escape Games were defined in the mathematical world. After finding out they were not, we also discovered it was an actual need[31]. Few literature exists on Escape Game, as the field of study is rather new. Mainly it is focused on creating a specific Escape Game and evaluating it. The same articles seem to serve as a basis for everyone and it looks like there are very few definitions of what an Escape Game actually is.

With the objectives of formalizing Escape Games, decision has been made to use Graph Theory. It turned out graphs suit pretty well the needs and the construct of an Escape Game. Defining the nodes and the update rules has been a long work, and probably flaws will be found here and there. Three graphs have been defined, each possessing their own node and edge types.

The Escape Game Graph represents a blueprint for a given Escape Game. It is the most important as the others can be derived from it. We attempted to reduce the node count as much as we could, but could not go below 6. In its original version, there were 3 edge types as well. In the current version, it seems rather complete. We hope it is a pace towards Escape Game formalization.

The Game Session Frame Graph is a snap shot of a real life game session and represent all the discovered items in a game at a given time. By itself, this graph doesn't mean much. Linked together by the Actions and Reaction mechanics, it brings an Escape Game alive.

The Game Session Graph is able to represent a full game session, from start to finish. To do so, Game Session Frame Graphs are linked together by player's actions and system's reactions. The Full Game Session Graph represent all the possible ways of playing the game. While very theoretical, it may provide usefulness.

Actions and reactions have been really difficult to build up, but we believe are the most interesting part. There are probably a few bugs and some of the functions might deserve a few corrections. However we trust they properly represent Escape Game mechanics.

When looking at annex B Game Session Illustration, the evolution of the Escape Game looks like a text-based RPG from the early computer times. In a way, we managed to create our own RPG game, with a small rule set.

A proper formalization is still required, as this work is only a step towards this goal. However it allowed us to partially answer the question : "Would it be possible to provide an escape game mathematical definition that includes the main aspects of an Escape Game and provide its evolution rules ?". The answer is that is it probably possible. That this work is an attempt at it. It still needs to be tested, and probably updated in order to be validated.

List of Tables

List of Figures

Bibliography

[1] 13 rules for escape room puzzle design. https://thecodex.ca/13-rules-for-escape-room-puzzle-design/. Last Accessed : 2023-06-25.

[2] Game theory. https://en.wikipedia.org/wiki/Game_theory. Last Accessed: 2022-05-22.

[3] List of games in game theory. https://en.wikipedia.org/wiki/List_of_games_in_game_theory. Last Accessed: 2022-05-22.

[4] Wilfried Admiraal, Jantina Huizenga, Sanne Akkerman, and Geert Dam. The concept of flow in collaborative game-based learning. *Computers in Human Behavior*, 27:1185–1194, 05 2011.

[5] Monica Arnal-Palacián, Juan Antonio Macías-García, and Isabel Duarte Tosso. Escape rooms as a way to teach magnitudes and measure in degrees in education. 03 2019.

[6] Luca Botturi and Masiar Babazadeh. Designing educational escape rooms: validating the star model. *International Journal of Serious Games*, 7(3):41–57, 2020.

[7] Neysa Brown, Wendy Darby, and Helen Coronel. An escape room as a simulation teaching strategy. *Clinical Simulation in Nursing*, 30:1–6, 2019.

[8] Ming Ming Chiu and Sze Wing Kuo. Social metacognition in groups: benefits, difficulties, learning and teaching. In *Metacognition: New research developments*, pages 117–136. Nova Science Publishers, Inc, 2009.

[9] Adam Clare. *Escape The Game: How to make puzzles and escape rooms*. Smashwords Edition, 2016.

[10] Samantha Clarke, Daryl Peel, Sylvester Arnab, Luca Morini, Helen Keegan, and Oliver Wood. Escaped: A framework for creating educational escape rooms and interactive games to for higher/further education. *International Journal of Serious Games*, 4, 09 2017.

[11] Stephany Coffman-Wolph, Kimberlyn Gray, and Marcia Pool. Designing an escape room game to develop problem solving and spatial reasoning skills. 2018.

[12] Clay Craig, Prisca S Ngondo, Michael Devlin, and Jennifer Scharlach. Escaping the routine: Unlocking group intervention. *Communication Teacher*, 34(1):14–18, 2020.

[13] Mihály Csíkszentmihályi. *Creativity: Flow and the Psychology of Discovery and Invention*. HarperCollins, 1996.

[14] Panagiotis Fotaris and Theodoros Mastoras. Escape rooms for learning: A systematic review. 10 2019.

[15] The Escape Game. The history of escape rooms (2021). https://theescapegame.com/blog/the-history-of-escape-rooms/, 2021. Last accessed: 19/05/2022.

[16] Bernard Guerrien. A review on use of escape rooms in education – touching the void. *real-world economics review*, issue no. 83:34–44, 03 2018.

[17] Robert A Hearn and Erik D Demaine. *Games, puzzles, and computation*. CRC Press, 2009.

[18] Outi Heikkinen and Julia Shumeyko. Designing an escape room with the experience pyramid model. 2016.

[19] Anne M Ho. Unlocking ideas: Using escape room puzzles in a cryptography classroom. *Primus*, 28(9):835–847, 2018.

[20] Ya-Hui Hsieh, Yi-Chun Lin, and Huei-Tse Hou. Exploring the role of flow experience, learning performance and potential behavior clusters in elementary students' game-based learning. *Interactive Learning Environments*, 24:1–16, 12 2013.

[21] Raph Koster. *Theory of fun for game design.* " O'Reilly Media, Inc.", 2013.

[22] Andrey Krekhov, Katharina Emmerich, Ronja Rotthaler, and Jens Krueger. Puzzles unpuzzled: Towards a unified taxonomy for analog and digital escape room games. *Proc. ACM Hum.-Comput. Interact.*, 5(CHI PLAY), oct 2021.

[23] Ángela Gómez López. The use of escape rooms to teach and learn english at university. *Research, Technology and Best Practices in Education*, pages 94–101, 2019.

[24] Alice Millot. *échappe-toi de la tour aux dragons.* éditions Milan, oct 2018.

[25] Scott Nicholson. Peeking behind the locked door: A survey of escape room facilities. 2015.

[26] Scott Nicholson. Ask why: Creating a better player experience through environmental storytelling and consistency in escape room design. *Meaningful Play*, pages 1–17, 2016.

[27] Rui Pan, Henry Lo, and Carman Neustaedter. Collaboration, awareness, and communication in real-life escape rooms. In *Proceedings of the 2017 conference on designing interactive systems*, pages 1353–1364, 2017.

[28] Ioannis Papadopoulos and Eirini Tenta. Escape rooms as a collaborative problem-solving environment. *International Journal of Game-Based Learning (IJGBL)*, 11(4):57–71, 2021.

[29] Katriina Penttilä. *History of Escape Games.* PhD thesis, Master's Thesis of University of Turku, 2018.

[30] Robert Ross. Design of an open-source decoder for educational escape rooms. *IEEE Access*, 7:145777–145783, 2019.

[31] Lene Taraldsen, Frode Haara, Mari Skjerdal Lysne, Pernille Jensen, and Eirik Jenssen. A review on use of escape rooms in education – touching the void. *Education Inquiry*, pages 1–16, 12 2020.

[32] Sanna Tarssanen and Panu Räisänen. *Handbbok for experience stagers.* Lapland Center of Expertise for the Experience Industry, 2009.

[33] Alice Veldkamp, Johanna Niese, Martijn Heuvelmans, Marie-Christine Knippels, and Wouter van Joolingen. You escaped! how did you learn during gameplay? *British Journal of Educational Technology*, 02 2022.

[34] M Wiemker, E Elumir, and A Clare. Escape room games: Can you transform an unpleasant situation into a pleasant one?. game based learning, 55, 2015.

[35] M Wiemker, E Elumir, and A Clare. Escape room games. *Game Based Learning. Vienna: Morawa Lesezirkel GesmbH*, pages 55–68, 2017.

A Articles by year query data

The following table contains data used in order to generate the graphics from Figure 1.1. The search query on Google Scholar was “"Escape Game" OR "Escape Room" OR "Exit Game" OR "Exit Room"” and has been adapted to the other platforms in order to perform the same research. This work has been performed on May 2, 2022.

Year	Google Scholar	ScienceDirect	Springer	ACM Digital Library	DBLP
1990	10	-	1	-	-
1991	9	-	0	-	-
1992	13	-	3	-	-
1993	14	-	2	-	-
1994	17	-	1	-	-
1995	22	-	5	-	-
1996	11	-	1	-	-
1997	18	4	3	-	-
1998	20	2	3	-	-
1999	18	2	2	1	-
2000	14	2	1	0	-
2001	25	1	1	1	-
2002	29	3	1	0	-
2003	47	3	5	1	-
2004	33	1	2	0	-
2005	48	2	4	2	-
2006	52	3	6	1	-
2007	69	7	4	2	1
2008	72	7	5	4	1
2009	58	11	5	0	0
2010	84	7	9	0	2
2011	83	5	13	1	2
2012	90	4	7	0	2
2013	134	3	11	2	2
2014	147	12	22	4	1
2015	181	6	16	3	1
2016	337	13	17	8	3
2017	537	19	16	8	3
2018	1120	28	38	15	5
2019	1530	31	68	17	9
2020	2100	55	107	30	7
2021	2350	82	154	34	7

B Game Session Illustration

This annex describes an Escape Game played from start to end by 2 players. The Escape Game is the one presented as example in Figure 3.1. Since there is only one starting position, both players will start in the first room. Figure B.1 is the Full Escape Game, with the skills. Some skill nodes have been cloned in order to keep the graph clean. Wherever two nodes skills have the same name, they are in fact the same node.

B.1 A game session

Figures B.2, B.3 and B.4 illustrates a Game Session. Here are the actions performed by the players :

F_0 Players P_1 and P_2 want to enter the game, they possess the required skills.

F_1 Players have entered the game (from now on, skills won't be displayed), The start in room R_1.

F_2 P_1 searches the room, a door to R_2 appears

F_3 P_2 moves to room R_2

F_4 P_2 searches the room, clue $C_{1.2}$ appears

F_5 P_1 searches the room, clue $C_{2.1}$ appears

F_6 P_2 searches the room, clue $C_{1.1}$ appears

F_7 P_2 searches the room, it is possible to move back to R_1

F_8 P_2 searches the room, clue $C_{1.3}$ appears

F_9 P_1 searches the room, clue $C_{1.4}$ appears

F_{10} P_1 memorizes clue $C_{1.4}$

F_{11} P_2 memorizes clue $C_{1.1}$

F_{12} P_2 memorizes clue $C_{1.2}$

F_{13} P_2 memorizes clue $C_{1.3}$

F_{14} P_2 moves to room R_1

F_{15} P_1 teaches clue $C_{1.4}$ to P_2

F_{16} P_2 searches the room, puzzle Z_1 appears

F_{17} P_2 makes an attempt on puzzle Z_1, all conditions are met, it is a success. Reward $C_{3.1}$ appears in the room.

F_{18} P_1 picks up clue $C_{2.1}$

F_{19} P_2 picks up clue $C_{3.1}$

F_{20} P_1 moves to room R_2

F_{21} P_1 searches the room, puzzle Z_2 appears

F_{22} P_2 moves to room R_2

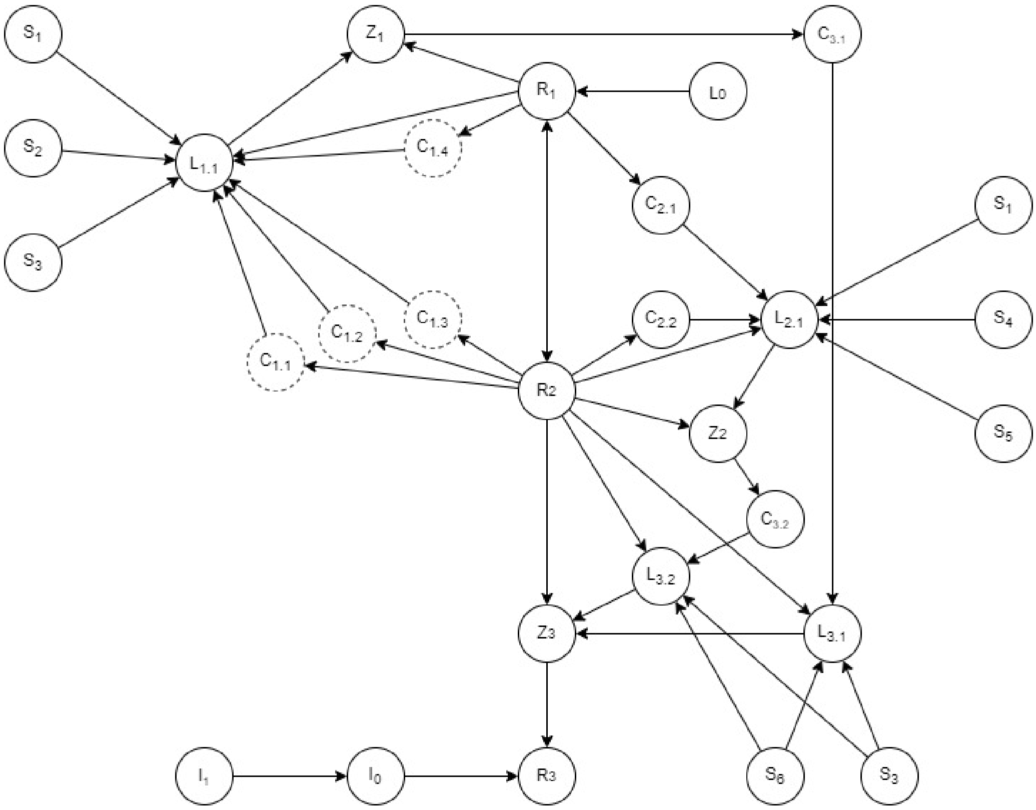


Figure B.1: The Full Escape Game Graph

F_{23} P_2 searches the room, clue $C_{2.2}$ appears

F_{24} P_1 picks up clue $C_{2.2}$

F_{25} P_2 searches the room, puzzle Z_3 appears

F_{26} P_1 makes an attempt on puzzle Z_2, all conditions are met, it is a success. Reward $C_{3.2}$ appears in the room.

F_{27} P_2 picks up clue $C_{3.2}$

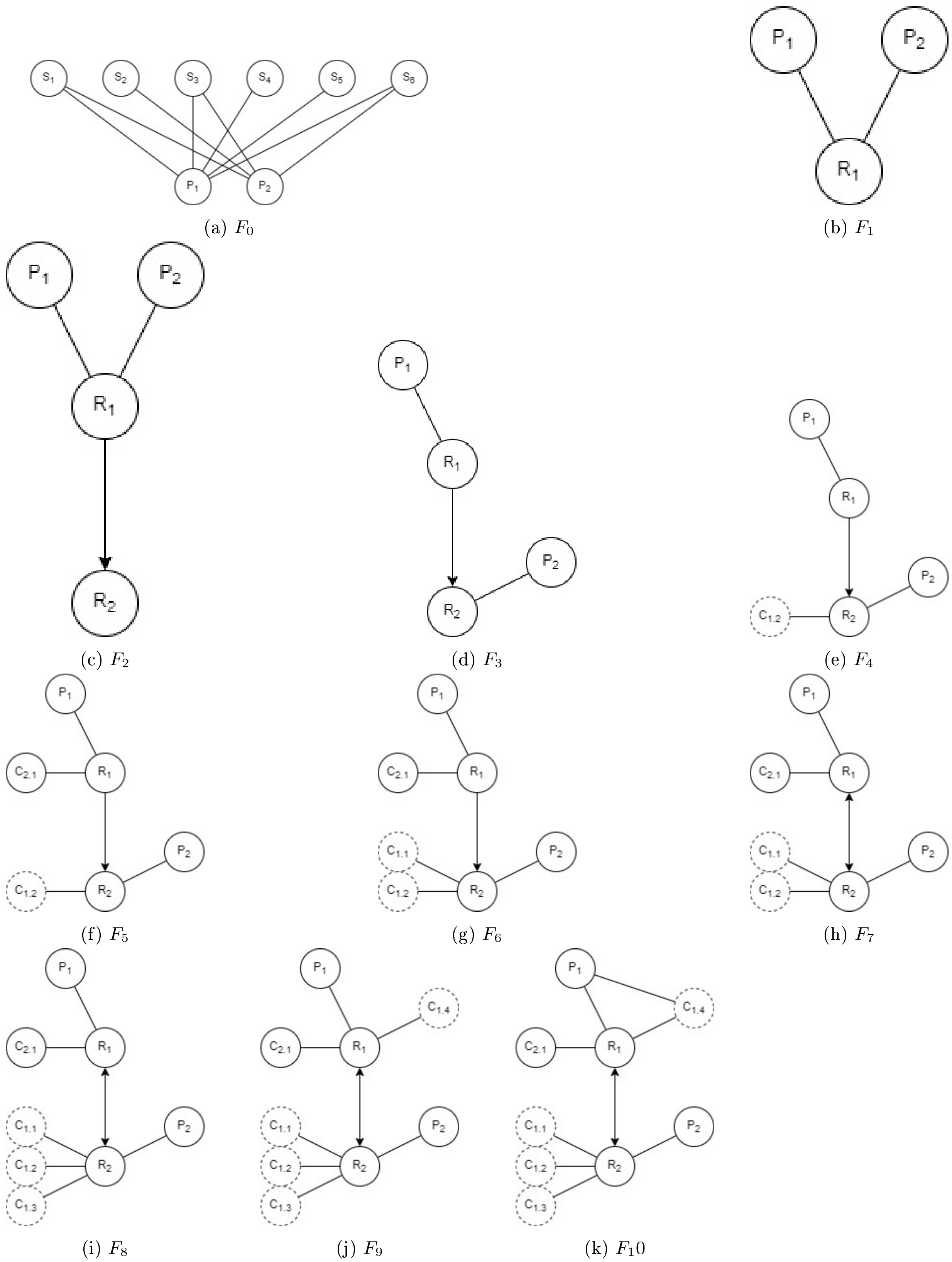
F_{28} P_2 gives clue $C_{3.1}$ to P_1

F_{29} P_1 and P_2 make an attempt on puzzle Z_3, all conditions are met, it is a success. A door opens to room R_3

F_{30} P_1 moves to room R_3, he exits the game.

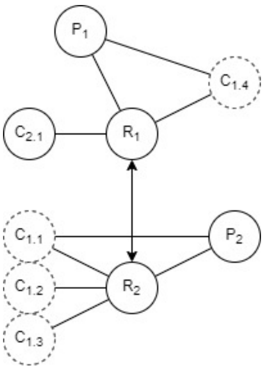
F_{31} P_2 moves to room R_3, he exits the game.

- The game is won !

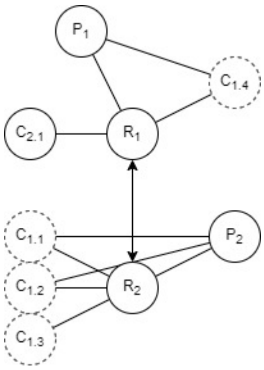


(a) F_0 (b) F_1 (c) F_2 (d) F_3 (e) F_4 (f) F_5 (g) F_6 (h) F_7 (i) F_8 (j) F_9 (k) F_10

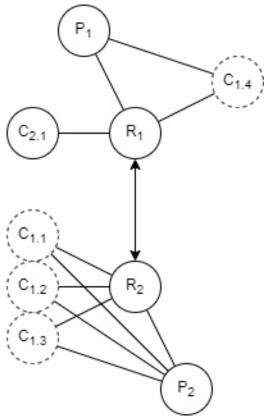
Figure B.2: A full Game Session (Part 1)



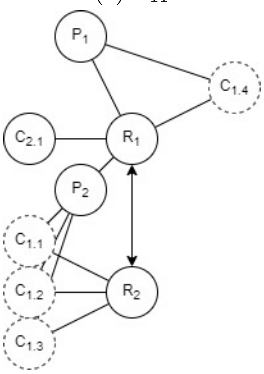
(a) F_{11}



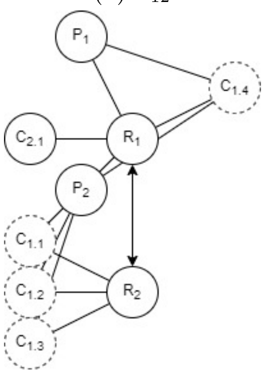
(b) F_{12}



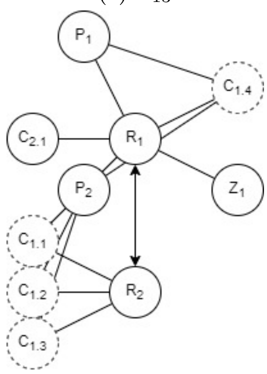
(c) F_{13}



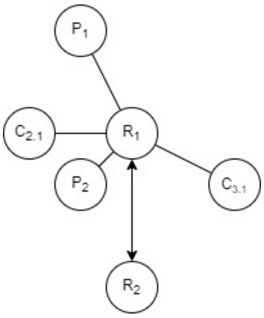
(d) F_{14}



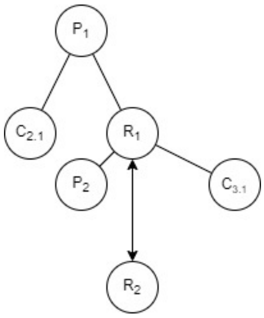
(e) F_{15}



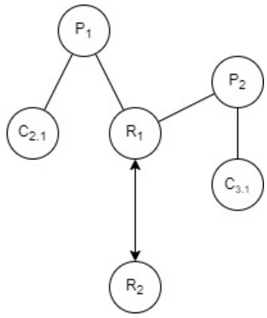
(f) F_{16}



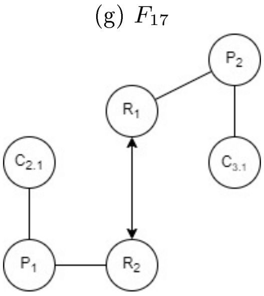
(g) F_{17}



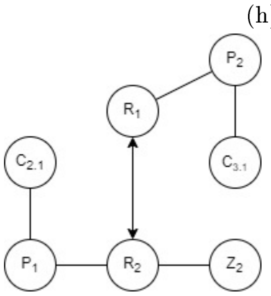
(h) F_{18}



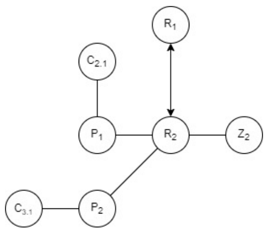
(i) F_{19}



(j) F_{20}

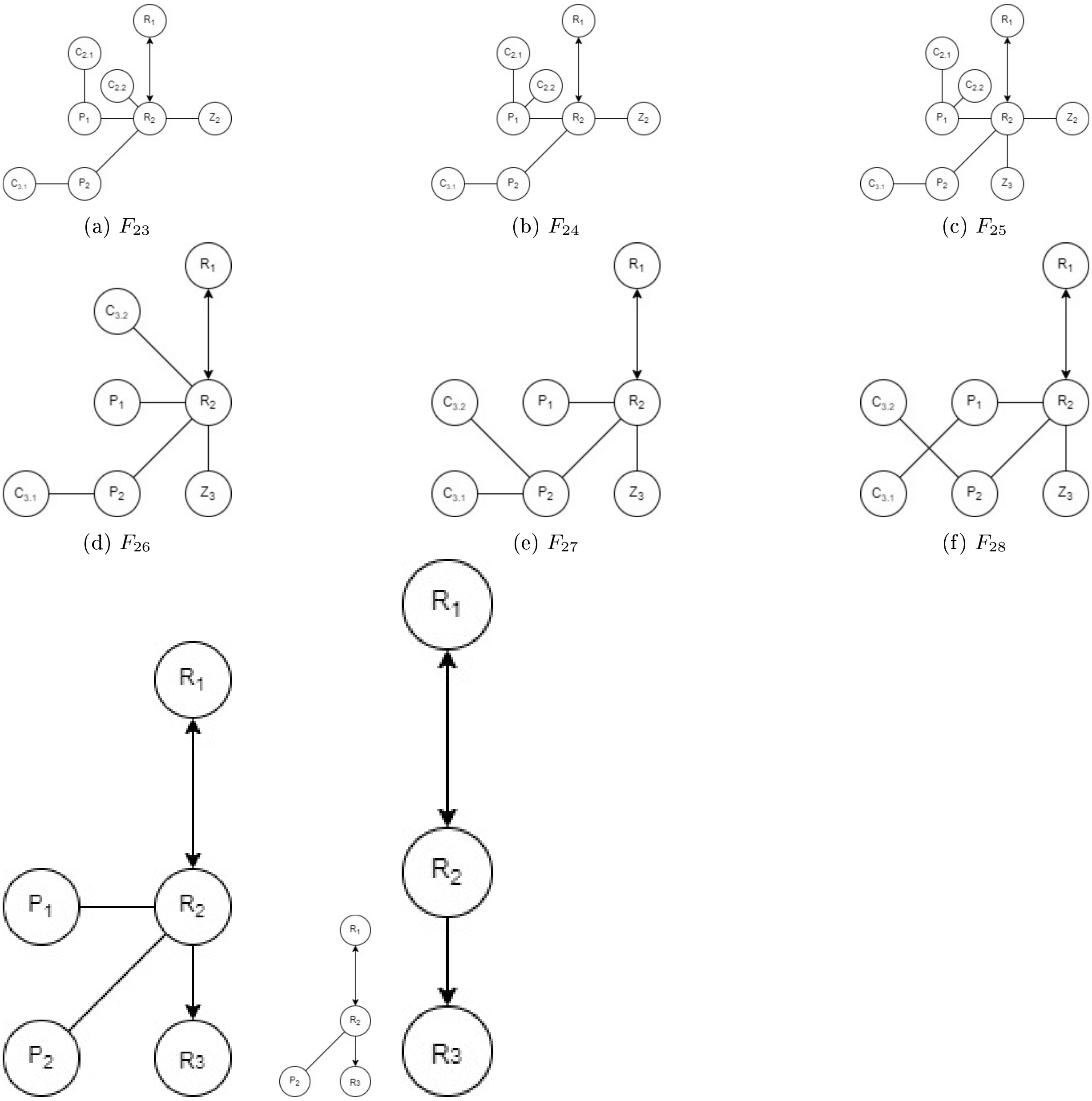


(k) F_{21}



(l) F_{22}

Figure B.3: A full Game Session (Part 2)



(a) F_{23} (b) F_{24} (c) F_{25}

(d) F_{26} (e) F_{27} (f) F_{28}

(g) F_{29} (h) F_{30} (i) F_{31}

Figure B.4: A full Game Session (Part 3)

An Escape Game Theory

Errata

p.7, § 6, instead of: “As of today, they exists”, read : “As of today, they exist”
p.9, § 5, instead of: “A previously stated”, read : “As previously stated”
p.12, 2.1.2, §1, instead of “course for nursing student”, read : “course for nursing students”
p.19, 3, §1, instead of “define Escape Game”, read : “define Escape Games”
p.20, 3.1.3, §1, instead of: “The following concept”, read: “The following concepts”
p.22, 3.3.2, §2: instead of: “taxonomy built by Krekhov and all”, read: “taxonomy built by Krekhov *et al.*”
p.25, 3.3.7, §2, instead of “That room represent”, read: “That room represents”
p.25, 3.3.7, §3, missing section reference, read : “(see section 3.4.6 Players)”
p.30, 3.4.6, §4, instead of : “Each player possess”, read : “Each player possesses”
p.31, 3.4.8, §2, instead of: “Game Session Frame represent”, read: “Game Session Frame represents”
p.31, 3.4.10, §1, instead of: “It represent”, read: “It represents”
p.31, 3.4.10, §2, instead of : “Edges (E_o) ”, read : “Edges(E_m) ”
p.32, 3.4.12, §2, instead of : “All the puzzle”, read : “All the puzzles”
p.35, next to last line, instead of: “M is a graph such a the one”, read: “M is a graph such as the one”
p.36, Attempt, instead of: “a player or a group of player”, read: “a player or a group of players”
p.39, §1, wrong formula. Instead of : “ (Find(Fk, ϕo, Location(Py) ∩ Cinfo)(Find(Fk, ϕo, Py) ∩ Cinfo) ”, read : “ (Find(Fk, ϕo, Location(Py)) ∩ Cinfo) \ (Find(Fk, ϕo, Py) ∩ Cinfo) ”
p.39, §1, wrong formula. Same reading as previous remark.
p.39, Communicate, §2, instead of: “represent”, read: “represents”
p.39, Communicate, §2, instead of: represent”, read: “represents”
p.40, §2, instead of: “only pick”, read: “only picks”
p.40, Function: Add, instead of: “added to game. The”, read: “added to game, the”
p.42, §1, instead of : “action has bee defined as”, read : “action has been defined as”
p.42, §6, last line, instead of: “illustrate”, read: “illustrates”
p.42, §7, remove “Since”
p.44, Connected Room, wrong formula. Instead of : “UndiscoveredRooms = (Py) = (Find(E, ϕ, Location(Py)) ∩ R)Rf”, read : “UndiscoveredRooms(Py) = (Find(E, ϕ, Location(Py)) ∩ R) \ Rf”
p.49, §2, instead of : “It possess”, read : “It possesses”
p.51, §2, instead of: “selected. Graphs”, read: “selected, graphs”
p.52, 4.2.2, instead of : “It would possible”, read : “It would be possible”
p.53, §4, instead of “represent”, read: “represents”
p.53, §5, instead of “represent”, read: “represents”
p.53, last §, instead of “provide”, read: “provides”