National Taiwan University of Science and Technology
Department of Computer Science and Information Engineering

以遊玩特徵為導向的程序化內容生成方法

Game Design Goal Oriented Approach for Procedural Content Generation

Ze-Hao Wang

Master Thesis Oral Defense

July 27, 2017

Prof. Wen-Kai Tai	Prof. A	Prof. B	Prof. C	Prof. D
Committee Chairperson	Committee Member	Committee Member	Committee Member	Committee Member

Agenda

- Introduction
- Related Works
  - Mission / Space framework
  - Map Sketches & Evolution of Segments
- Proposed Methodologies
  - System Architecture
  - Mission Grammars
  - Room Definitions and Instruction
  - Genetic Algorithm in Segments Evolution
- Experimental Results
- Conclusions and Contributions
- Future work

Introduction

Motivation

Research Goals

Motivation

- [會再翻成英文]
- 程序化內容自動生成 (Procedural Content Generation) 在過去就廣泛被應用於遊戲設計領域，其主要目的為增加遊戲內容的隨機性與多樣性。我們預期讓玩家在進行遊戲時能夠遵循關卡設計師的劇情脈絡外，亦能夠體驗到有意義且多樣化的遊戲關卡內容。
- 複雜系統

Dungeon Architect

DunGen

	Dungeon Architect	DunGen
Build topology	Randomly PCG, Manually	Randomly PCG
Mission-based topology	None	Stable-linear structure

Research Goals

- [會再翻成英文]
- 我們針對遊戲過程中的遊玩特徵 (gameplay patterns) 進行抽象化，使用程序化生成技術產生帶有意義遊戲關卡內容，藉此消彌或降低因隨機性所產生的不穩定要素，以改善並豐富遊戲體驗。

Our Method

	Our Method	Dungeon Architect	DunGen
Build topology	PCG	Randomly PCG, Manually	Randomly PCG
Mission-based topology	Dynamic-nonlinear, meaningful structure	None	Stable-linear structure
Game Patterns	Genetic Algorithm	None	None

Related Works

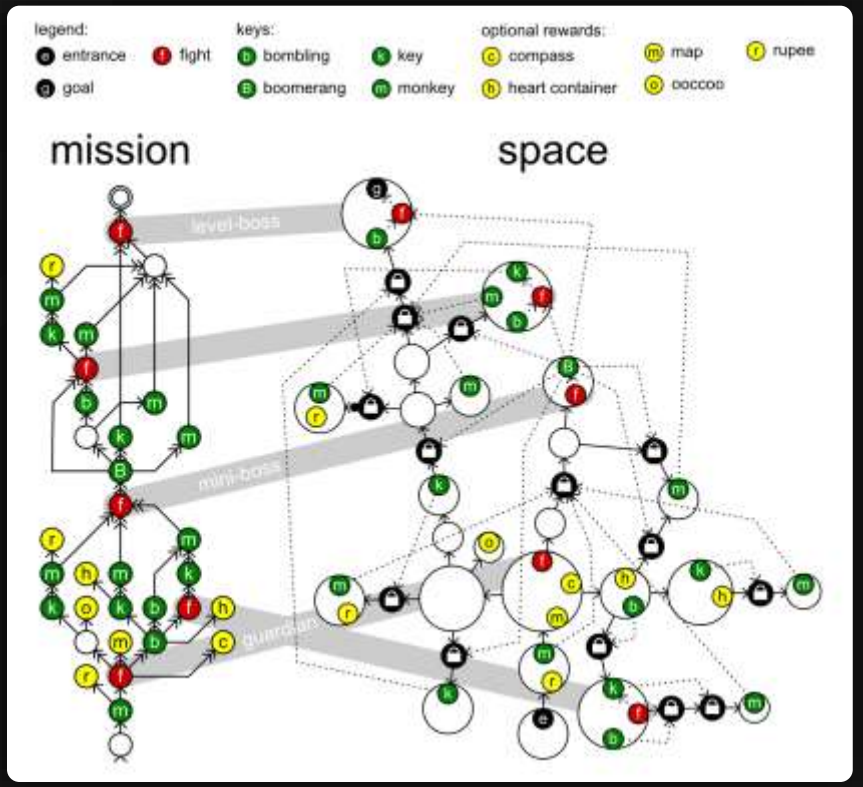
Mission / Space framework

Map Sketches & Evolution of Segments

Mission / Space framework

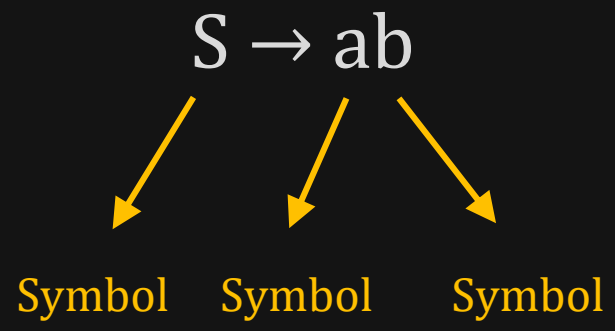
Mission/Space framework, focuses on **level design** and the **mechanics** that control player progression through a game.

- Transformational Grammars
- Mission Grammars
- Space Grammars
- Mission Graph Convert into Space Graph



Transformational Grammars

- Be consisted from **an alphabet** and **a set of rules**
  - **Alphabet**
    It is a set of symbols the grammar works with.
  - **Set of rules**
    It specifies what symbol can be replaced by what other symbols to form a new string.
- Sides and symbols of the rule
  - **Terminals** (common in lowercase)
    Symbols in the alphabet can never be replaced because there are no rules.
  - **Non-terminals** (common in uppercase)
    Symbols have rules that specify their replacement.



{ S, a, b } is an alphabet

Mission Grammars

-  **Inhibitions**

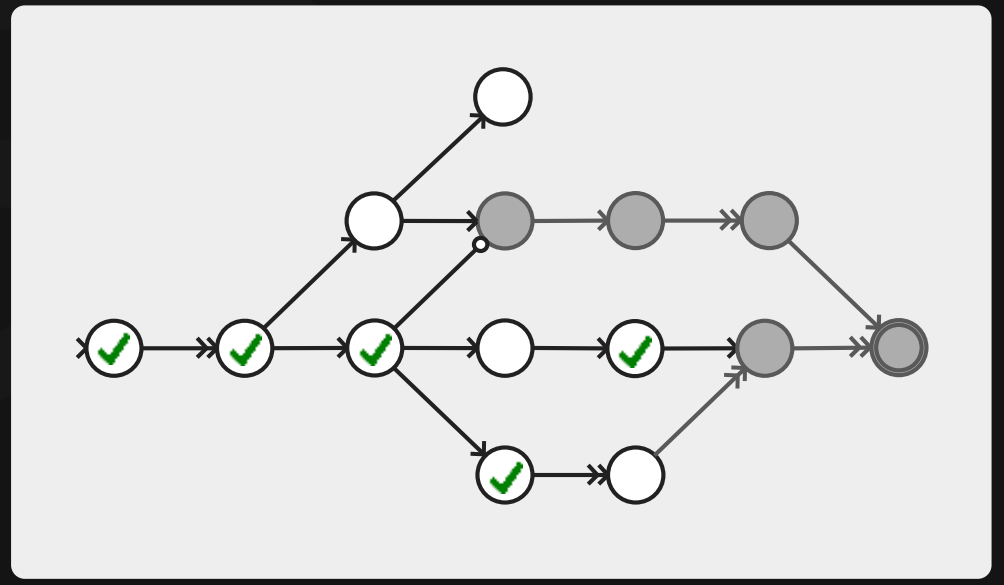
  Available when at least one of the weak prerequisites is completed.

-  **Strong requirement**

  Available when all strong prerequisites are completed.

-  **Weak requirement**

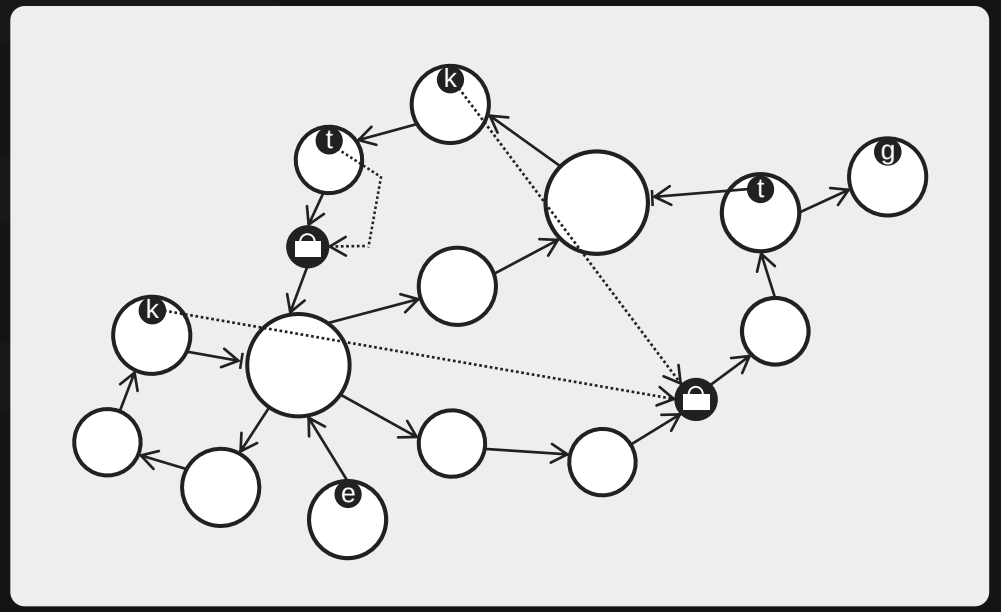
  Available when at least one of the weak prerequisites is completed.



Example of mission graph

- Mission graphs represent the players' progress towards a goal not by tracking their physical location, but by tracking the tasks they must perform to finish a level.
- A mission graph is a directed graph that represents a sort of to-do list with each node representing a task that might or must be executed by the players.

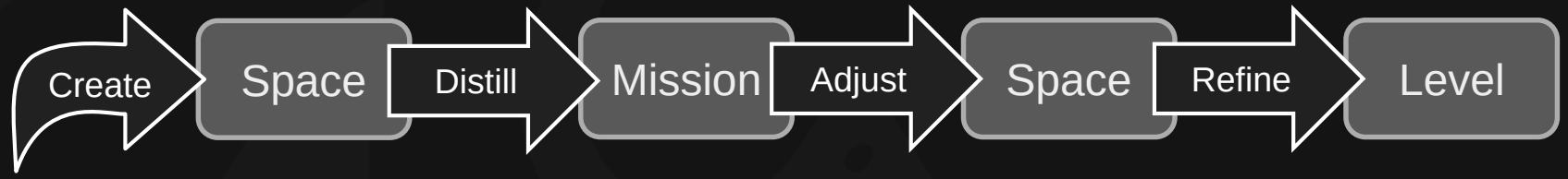
Space Grammars



Mission Graph Convert into Space Graph

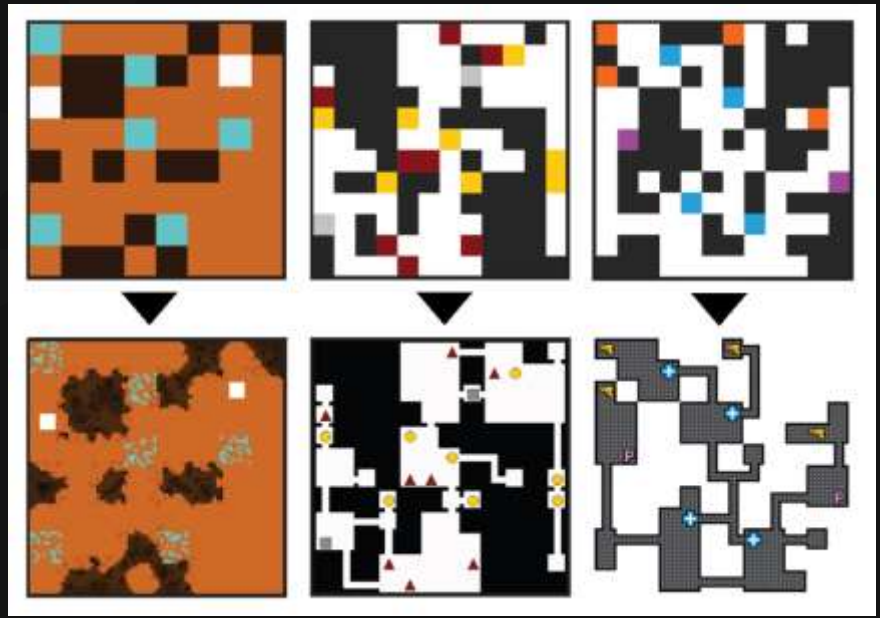
- The steps in the generation progress investigated in this paper in detail:

- An alternative method, does not go into detail:

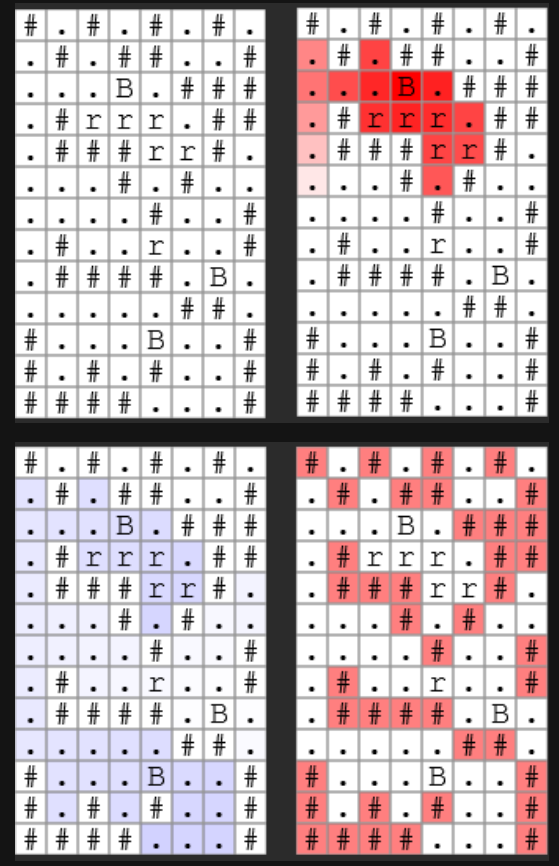


Map Sketches & Evolution of Segments

- Map Sketches
- Map Sketch Evolution
- Dungeon Segments
- Dungeon Segment Evolution

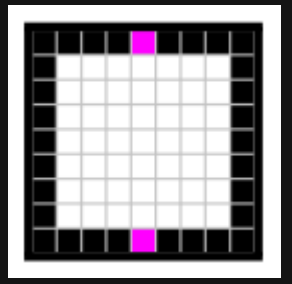


Map Sketches



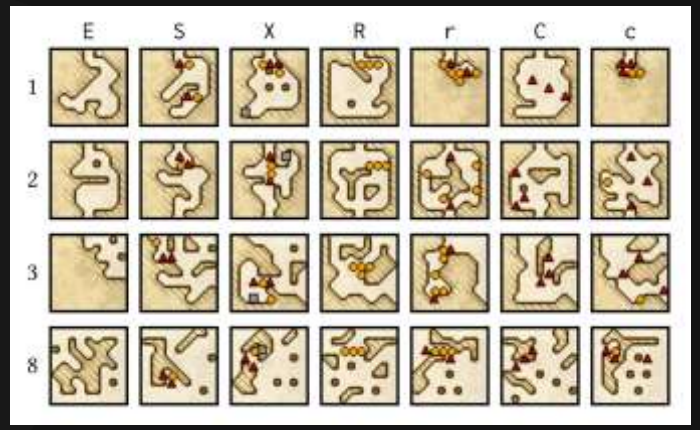
Map Sketch Evolution

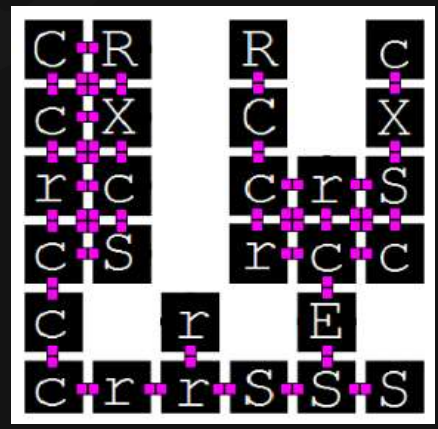
Dungeon Segments



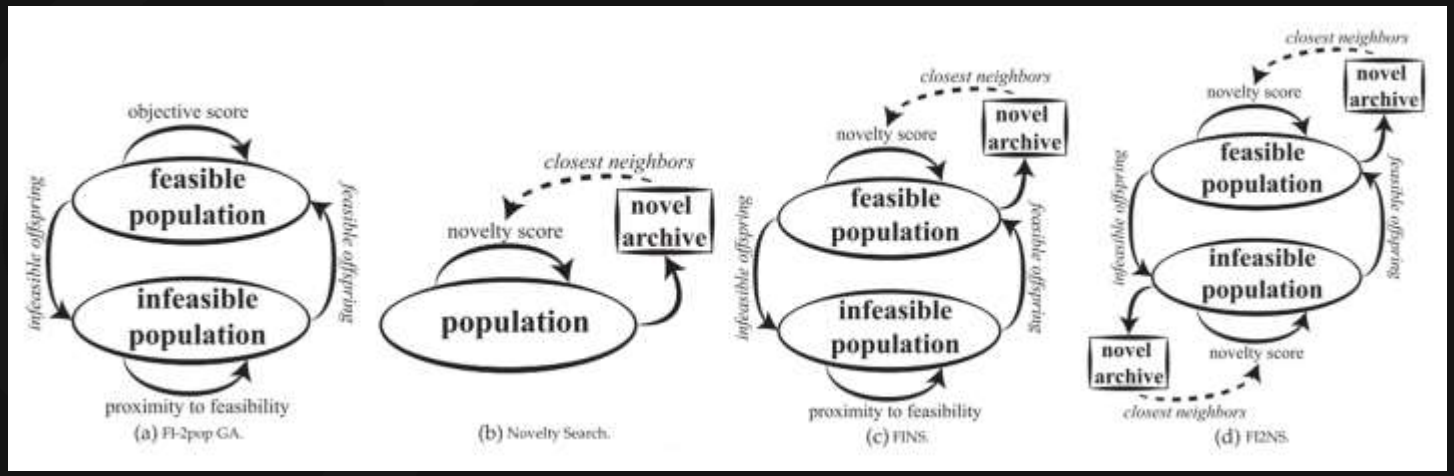
.	Empty	#	Wall
n	Connection	x	Exit
m	Monster	t	Treasure

Dungeon Segment Evolution





Constrained Novelty Search



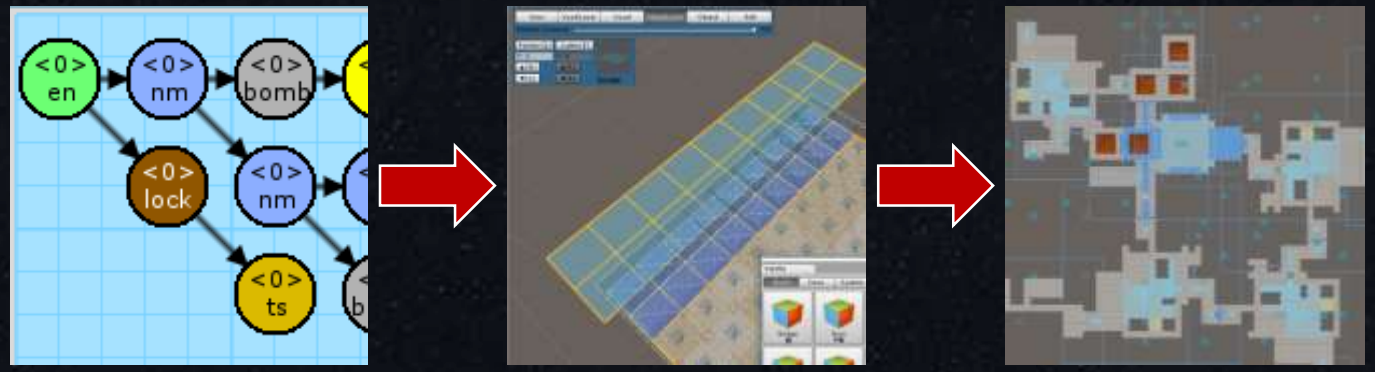
Proposed Methodologies

Motivation

Research Goals

Research Goals

System Architecture



- Phase 1 -

Mission / Space framework

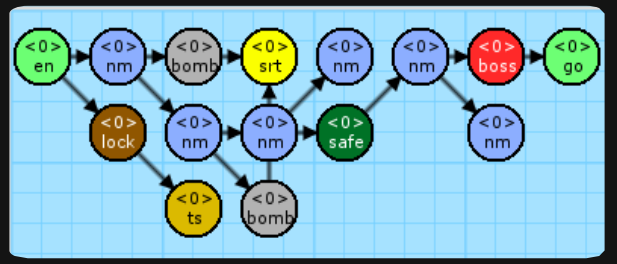
- Phase 2 -

Dungeon Segment Evolution

System Architecture

1. Create Mission Grammars

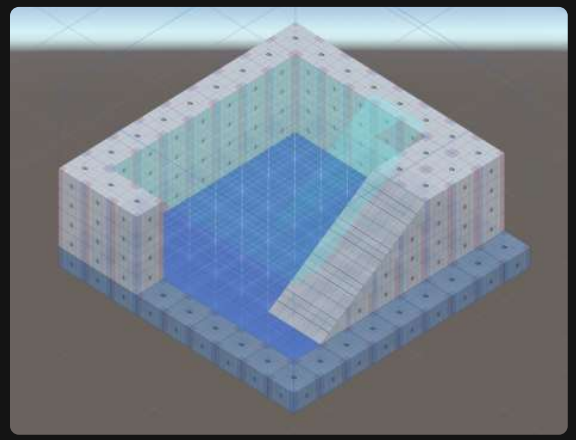
- Mission Alphabet
  - Terminal nodes & non-terminal nodes
- Mission Rules



System Architecture

2. Create Volumes

- Volex-based volumes



System Architecture

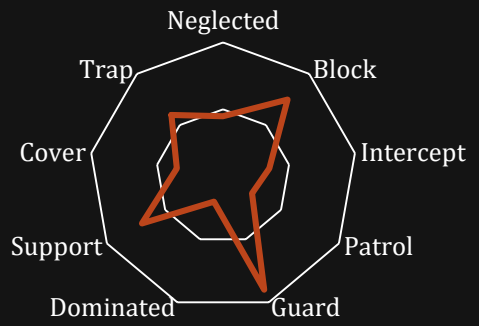
3. Generate the Space

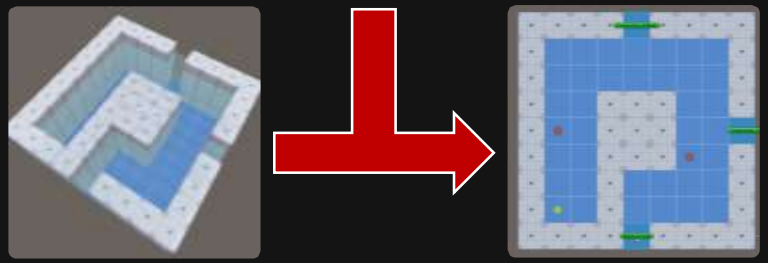
- Instruction of Rewrite System
- Replacement of Rewrite System

System Architecture

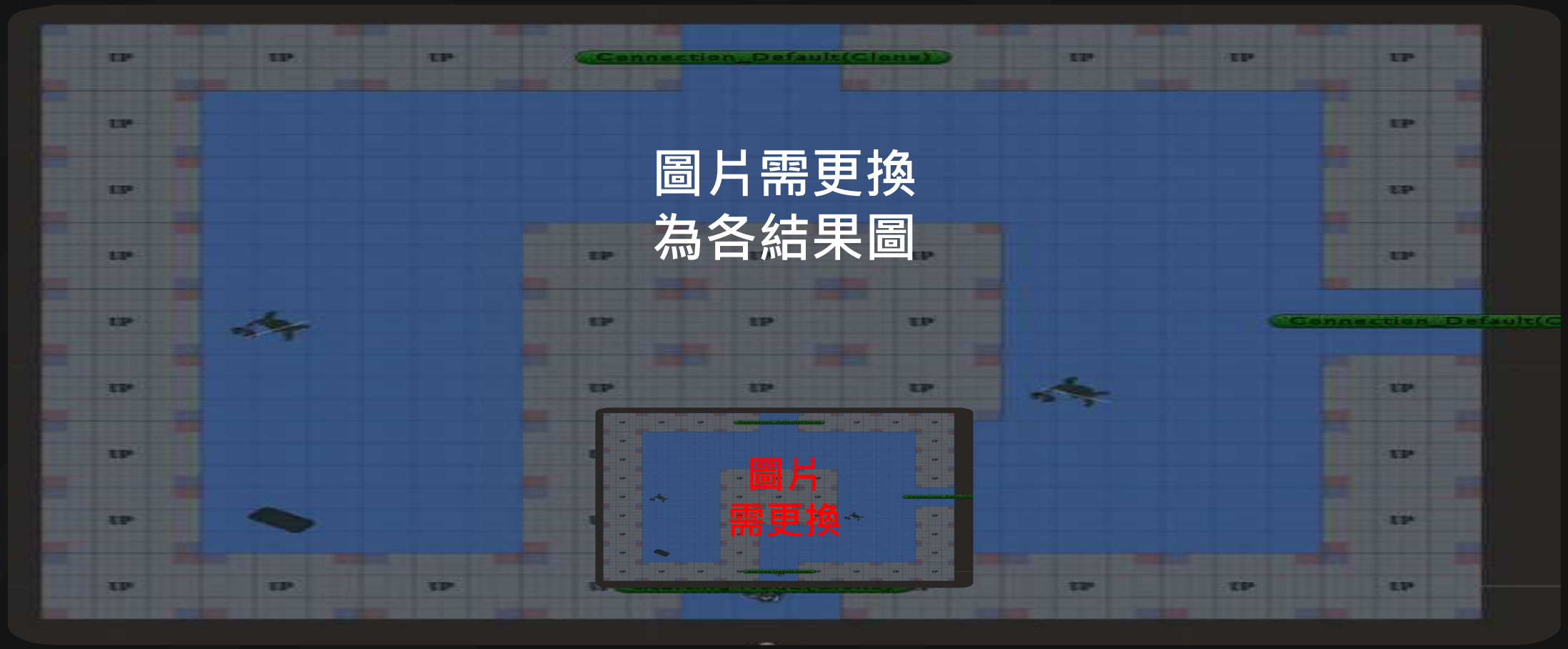
4. Generate the Game Patterns

- Genetic Algorithm for emergence objects
- Nine main metrics to design the fitness functions





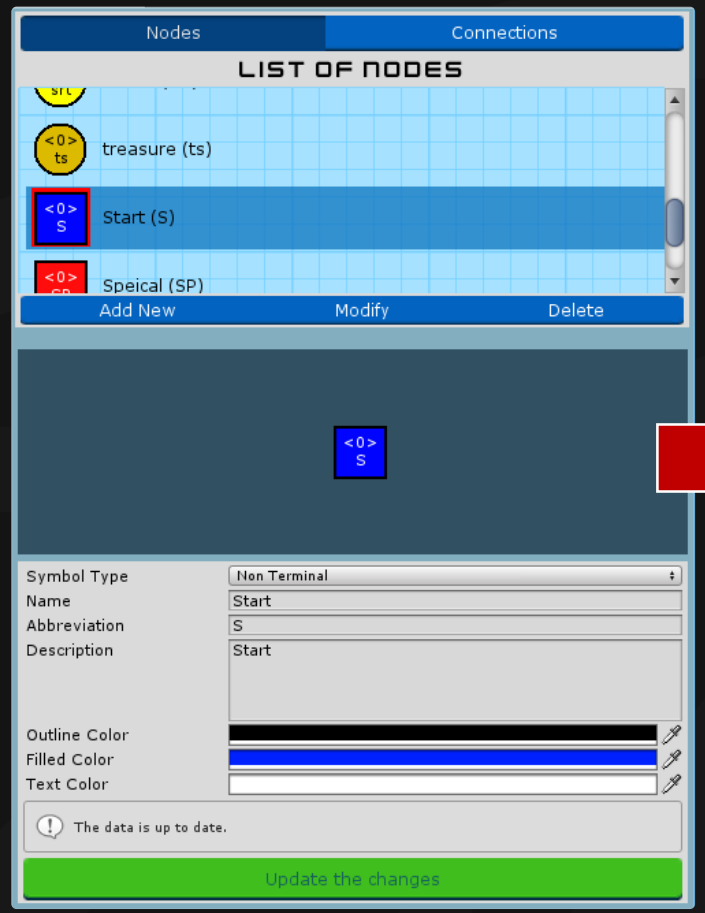
System Architecture



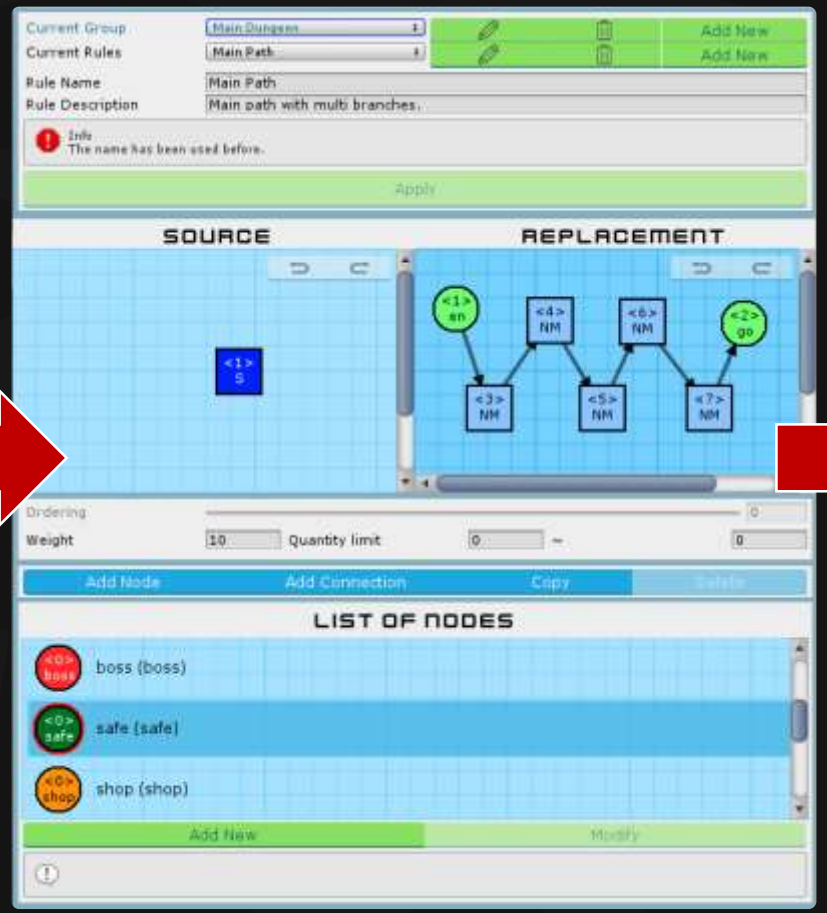
System DEMO

影片將會放置於此

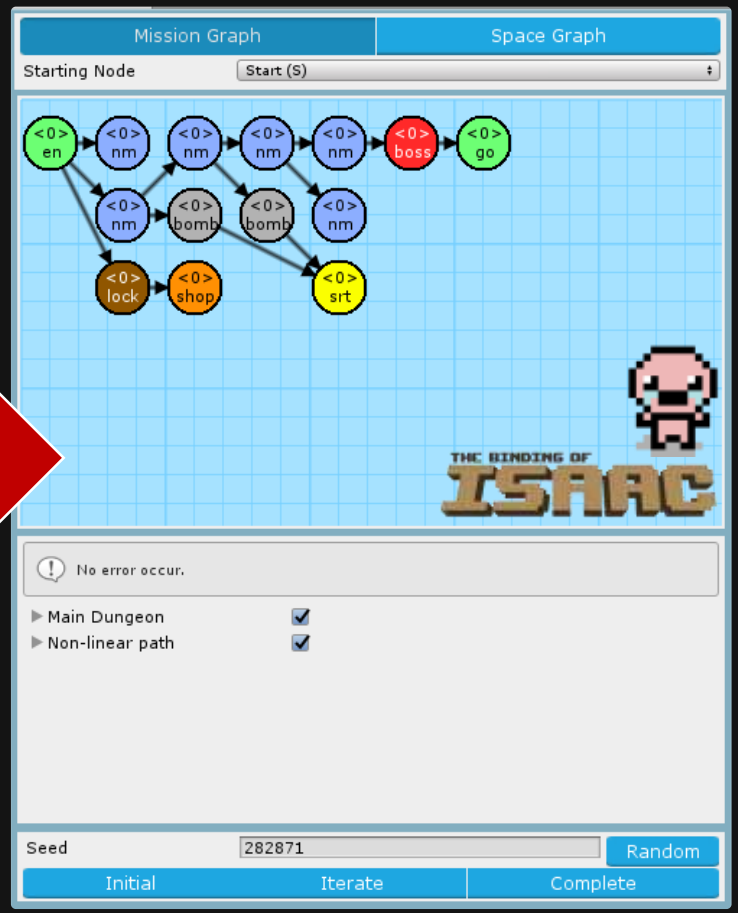
Mission Grammars



Mission Alphabet

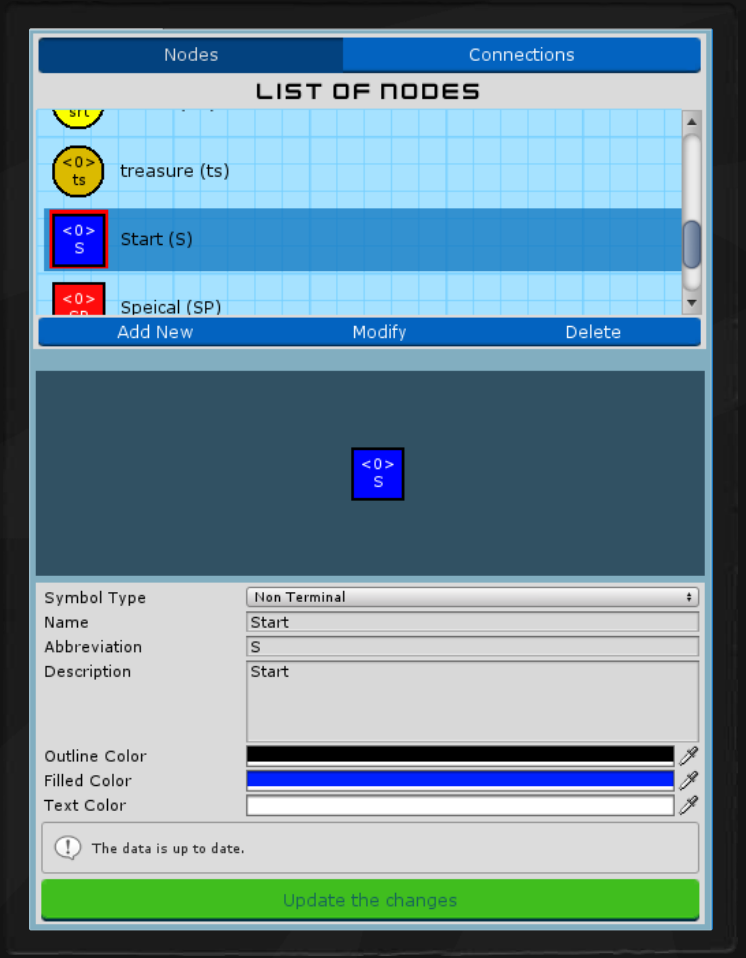


Mission Rules



Mission Graph

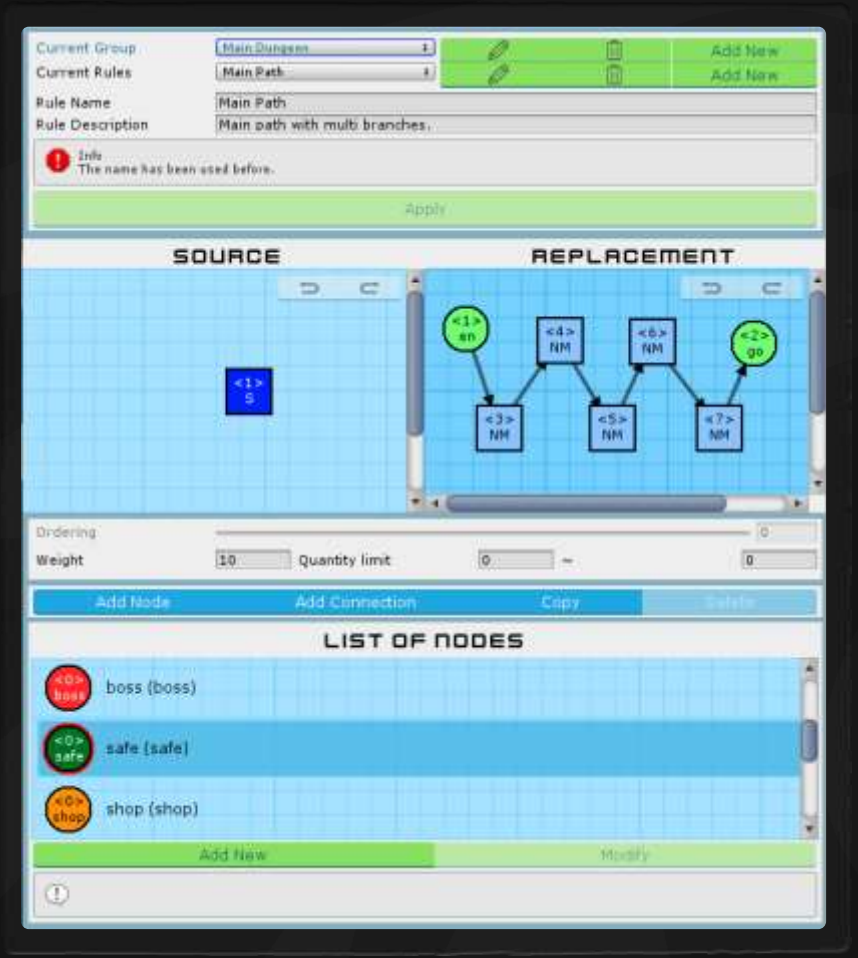
Introduction of Mission Alphabet



Mission Alphabet Window

- List of symbols
  - Nodes and connections
  - Directly preview before selected
- Preview area
  - Preview the symbol after editing immediately
  - Submit hint and form validations
- Default and extended nodes
  - Default: *None*, *Entrance*, *Goal*
  - Extended: *Any*

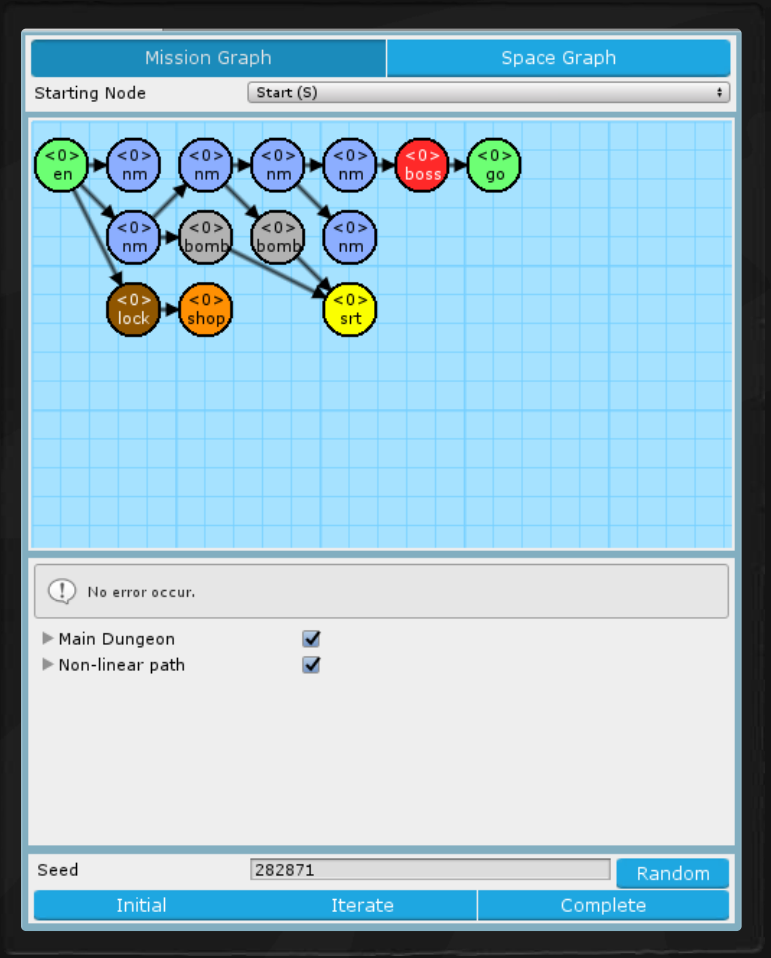
Introduction of Mission Rules



Mission Rules Window

- Hierarchy structure
  - Groups > Rules > Source & Replacement
  - Friendly interface to create, delete and edit.
- Rule canvas
  - Drag & drop to modify symbols
  - Custom size of canvas
  - Selected symbol highlighting
  - Connections stick on nodes automatically
  - Back trace the states (Redo / Undo)

Introduction of Mission Graph



Mission Graph Window

- Set starting node
  - The head of the mission graph
- Preview mission graph
- Rewrite system
  - Based on the mission grammar previous window set
  - Toggle the selectable rules
  - Find the match parts in graph with rules
    - VF Graph Isomorphism Algorithm

Usage flow

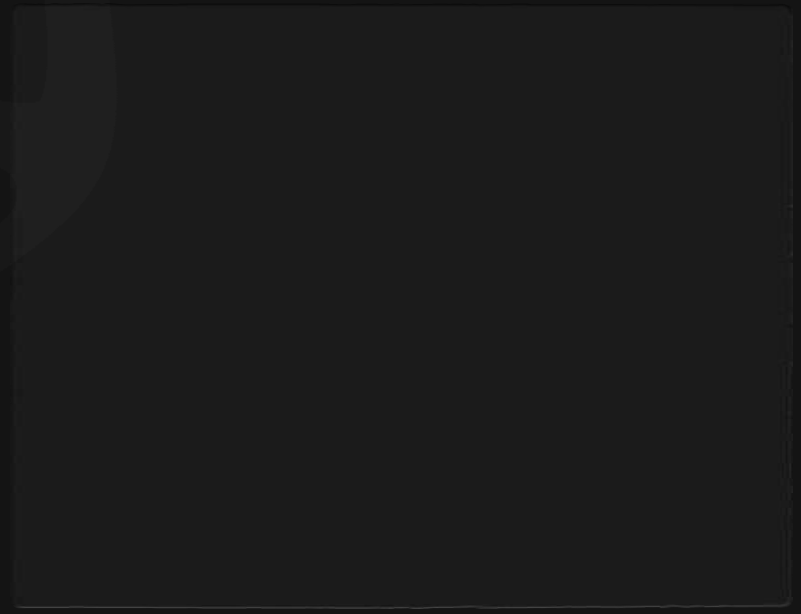
A Create the Mission Alphabet

Node	Start (S)	entrance (en)	Normal (NM)	normal (nm)	boss (boss)	goal (go)	safe (safe)	Special (SP)	shop (shop)	bomb (bomb)	lock (lock)	secret (srt)	treasure (ts)
Display													
Quantity limit	0	0	0	8	1	0	0	0	1	0	0	1	1
Require-ment					Battle and end			Multi - choice	Get the key from secret, then enter the secret or treasure room behind the wall that need to use the bomb to destroy it				

※ Limit of *Zero* Means infinite.

Usage flow

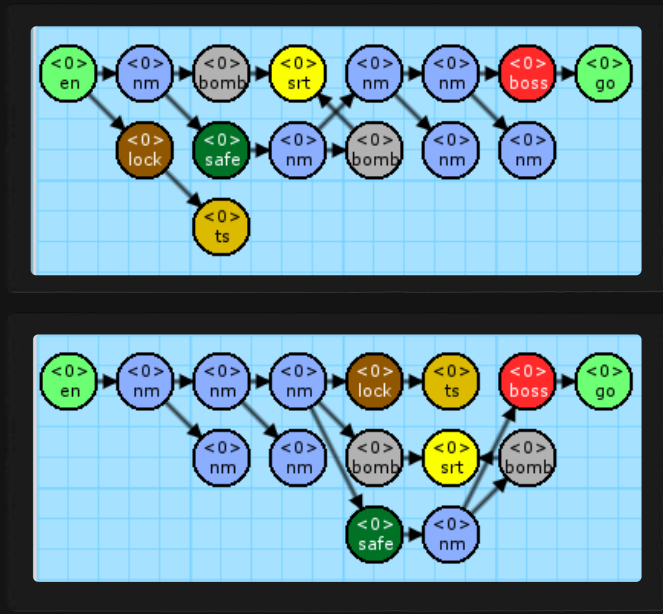
B Design the Mission Rules



Usage flow

C Export the Mission Graph

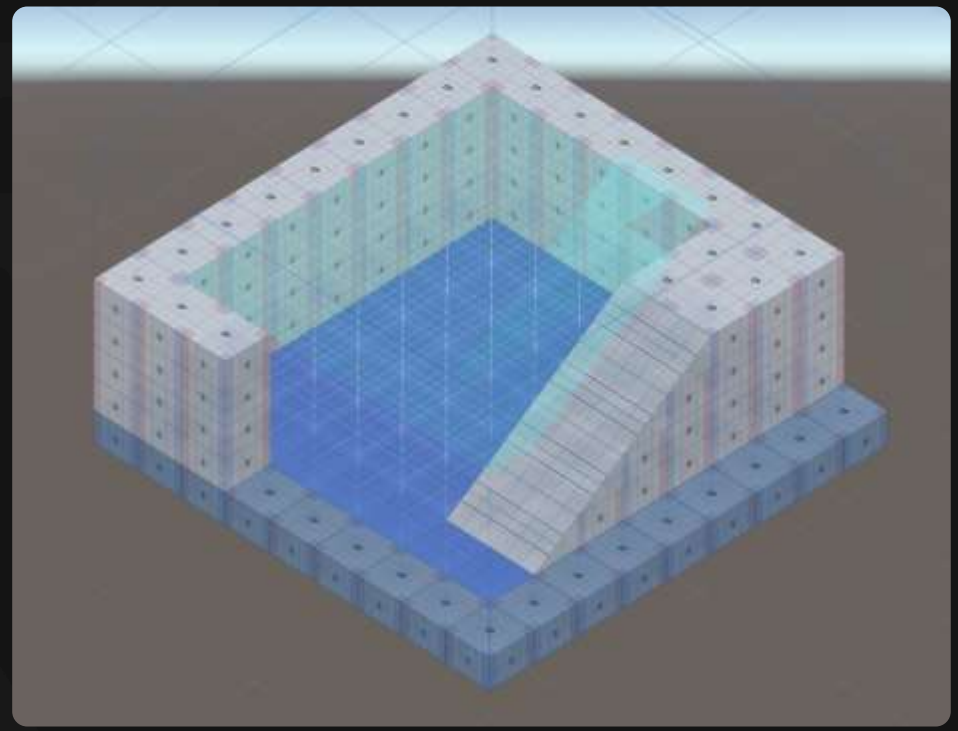
- Set the starting node
  - Start to execute the iteration from the root.
- Set the random seed
  - Seed will affect **selection** when there are many candidates, or and the **additional iterations** [1] in the moment.



[1] additional iterations: The completed generation means all nodes in the graph are replaced to terminal nodes. Additionally executes some rules to replace the current graph.

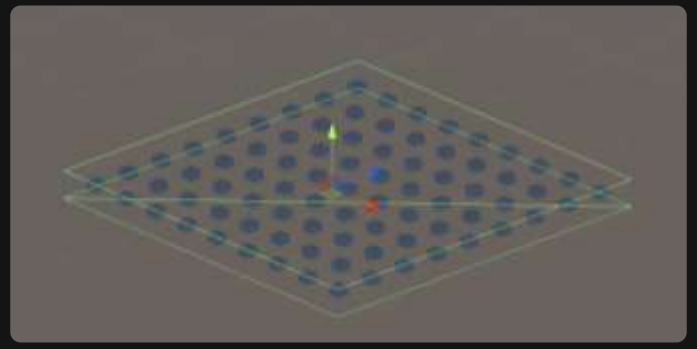
Room Definitions and Instruction

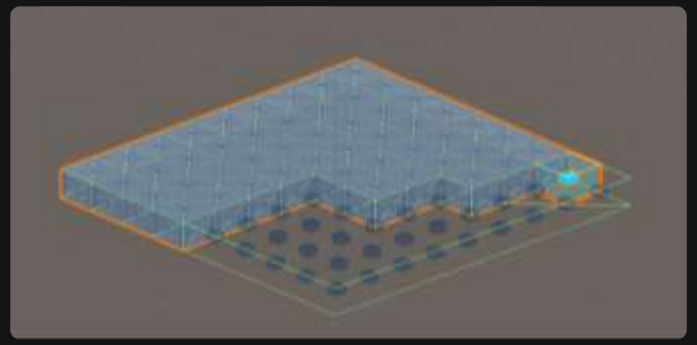
- Editor
  - Voxel-based units
  - Markers I - Decorations
  - Markers II - Connections
- Rewrite System
  - Instruction
  - Replacement



Voxel-based units in Editor

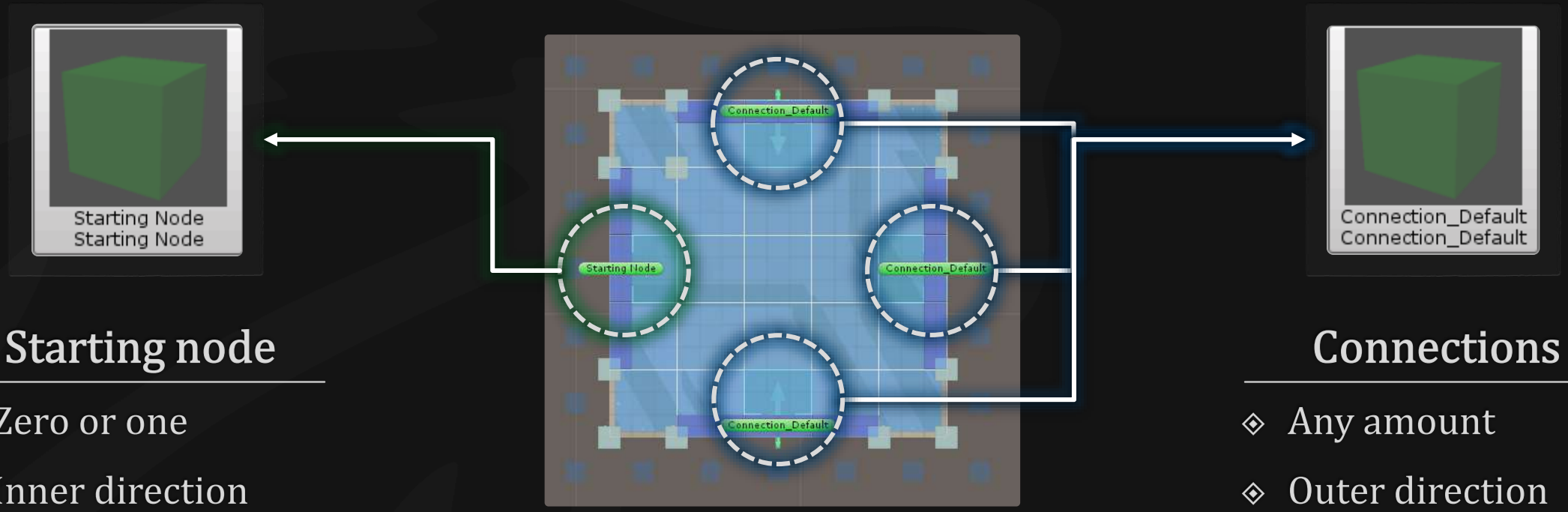
- Hierarchy
  - **Level**
    A set of volumes. Expresses a complete game level.
  - **Volume**
    A set of chunks. Mostly expresses a room.
  - **Chunk**
    Consists of 9x9x9 voxel. Based on the size of volume, uses different numbers of chunk.
  - **Voxel**
    The minimum unit of level, size is 3x2x3. It can be a block or nine direction decorations.





Decorations in Editor

Connections in Editor



Starting node

- Zero or one
- Inner direction

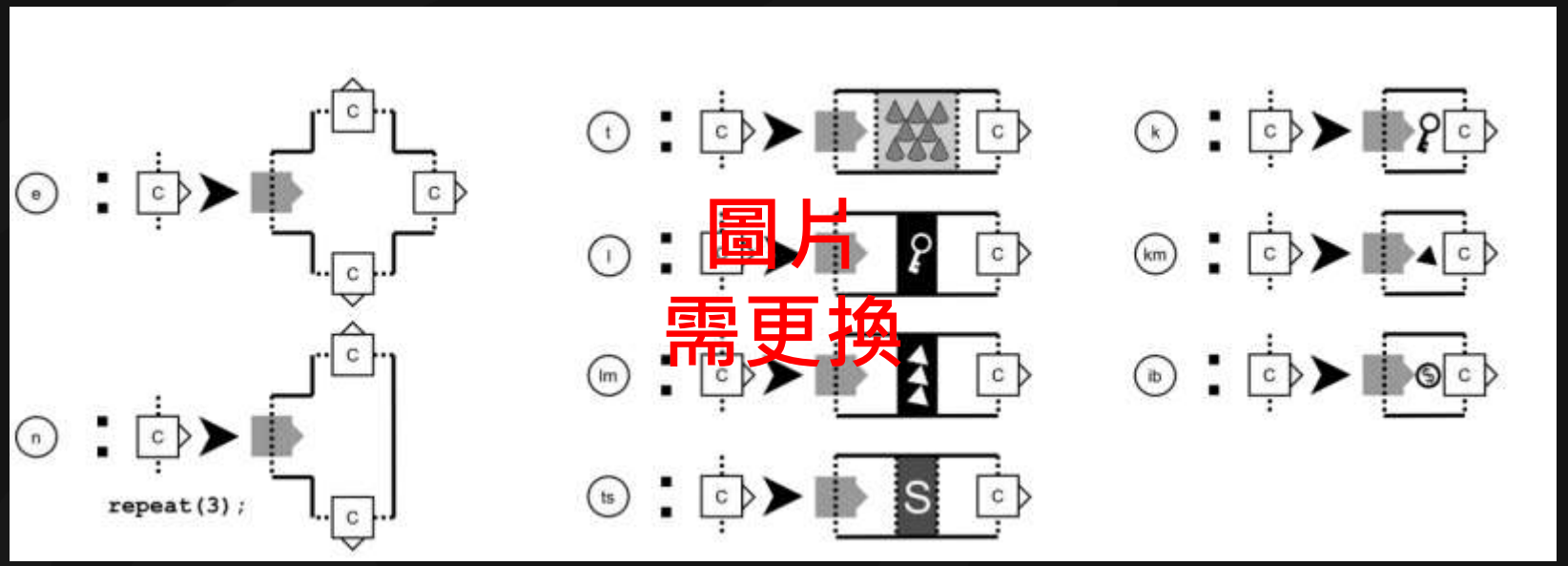
Connections

- Any amount
- Outer direction

- The **connection** of "**A volume**" will stick with the **starting node** of "**B volume**", according to their direction of arrow. If doesn't exist any stating node, will pick one connection randomly.

Instruction of Rewrite System

- Building Instructions
  - Each rule in the shape grammar was associated with a **terminal symbol** form in the mission grammar.
  - Look for rules that implement that symbol, selects one at random based on their **relative weight**.

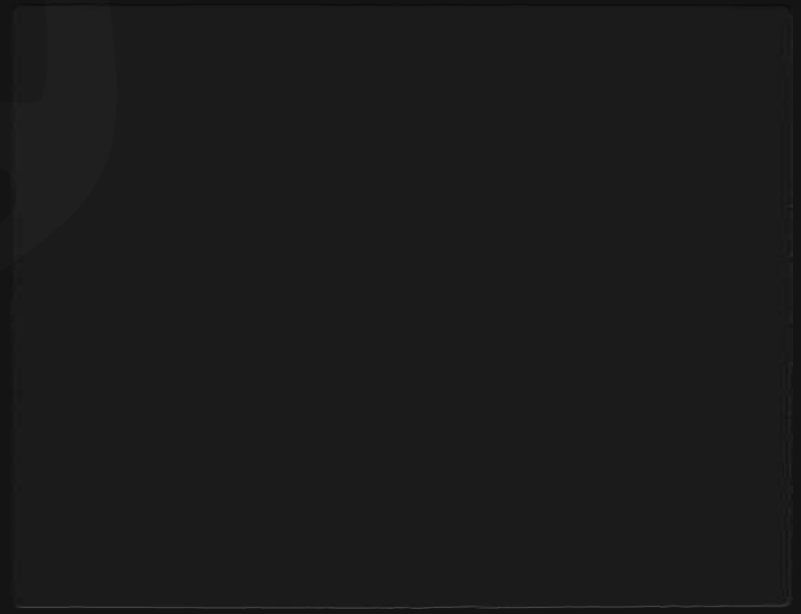


圖片需更換

Replacement of Rewrite System

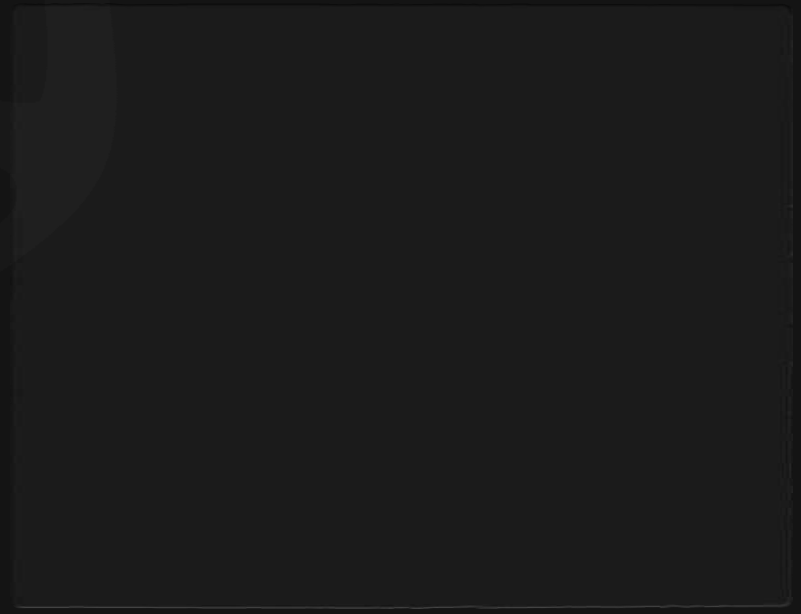
Usage flow

D Design the Volumes



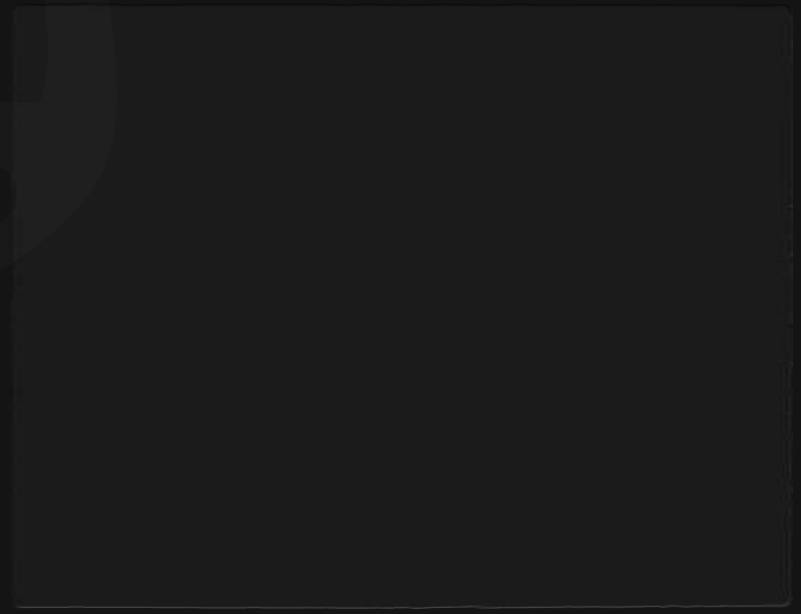
Usage flow

E Append the Starting Node & Connections



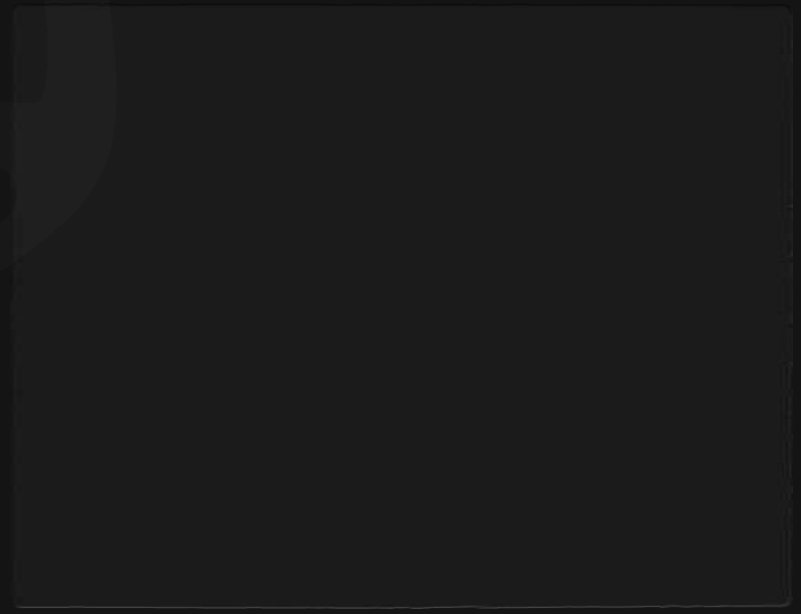
Usage flow

F Defined the Instruction of Rewrite System



Usage flow

- Defined the Replacement of Rewrite System



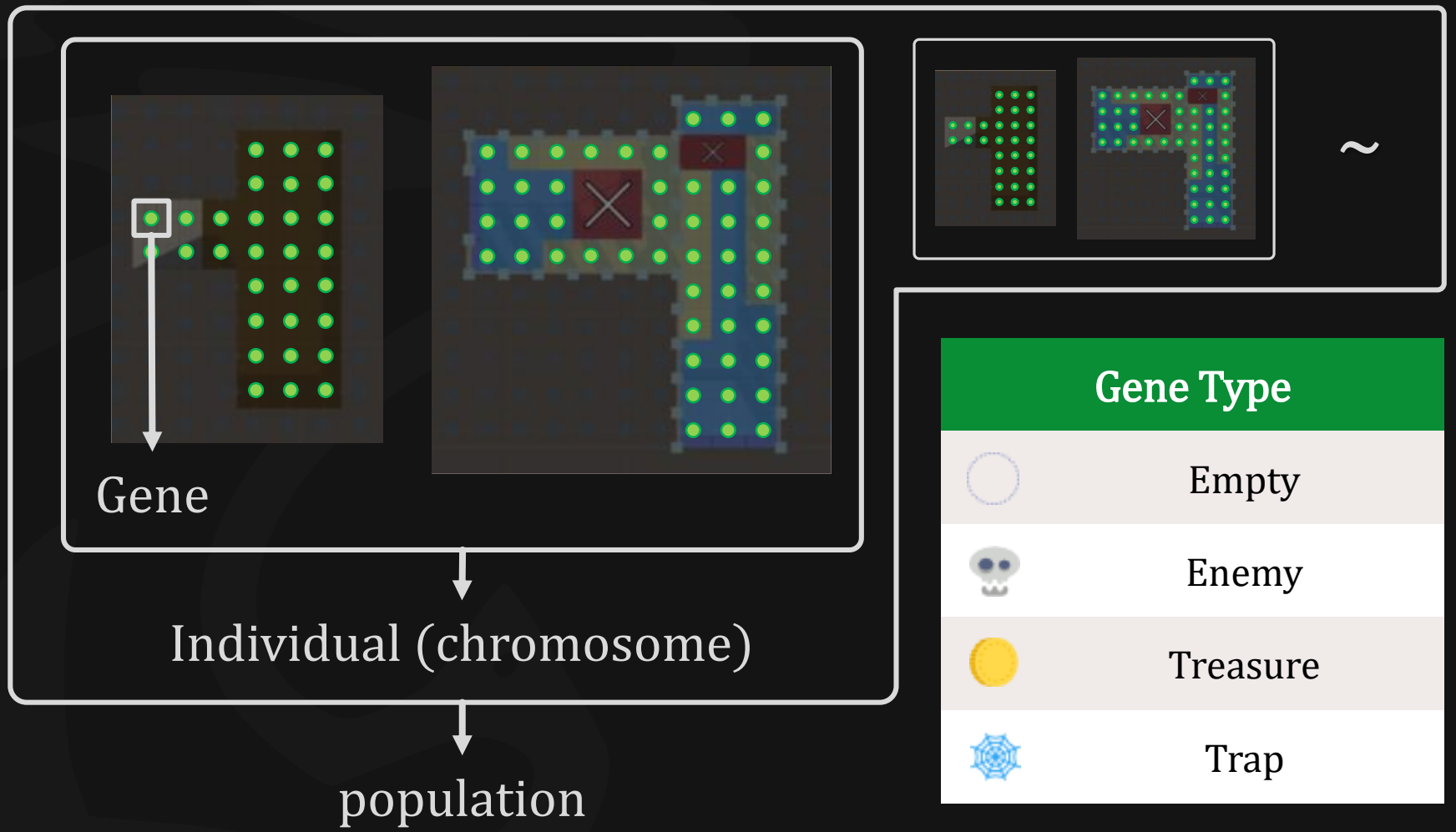
Genetic Algorithm in Segments Evolution

- Genetic Algorithm
  - Algorithm flow
  - Gene
  - Crossover
  - Mutation
- Nine fitness functions
  - Neglected
  - Block
  - Intercept
  - Patrol
  - Guard
  - Dominated
  - Support
  - Cover
  - Trap

Gene

Basement

Ground



Gene Type
Empty
Enemy
Treasure
Trap

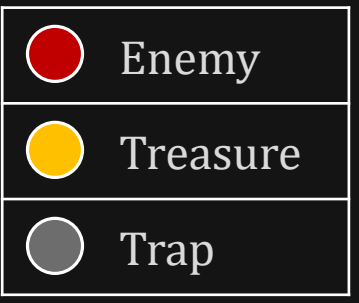
Crossover

Mutation

Nine fitness functions – 1st Neglected

- Fitness function:

- Description:
  [會再翻成英文]
  由於房與房之間的牆壁阻隔，使得敵人能夠埋伏於入口附近之死角處，出奇不意地對玩家展開攻擊。為了體現出這種現象，我們將敵人 (E_i) 與主要動線上各點 (MP_i)，兩端點連線之對角線所構成的立方體，立方體所涵蓋各座標點 (N_j) 至該對角線的距離為 d_k，隨著距離增加影響程度會衰減；vis_k 為該點的可視情形，若有不可視的座標存在便會提高適應值。隨著動線的順序演進，影響程度逐漸衰減。



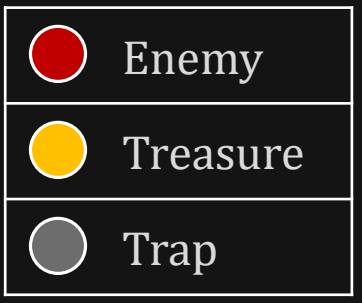
Example gameplay of "Neglected"

Nine fitness functions – 2nd Block

- Fitness function:

$$f_{blk} = \log_{MP} E \,, MP = \sum_{i=1}^{N} mp_i \,, E = \sum_{j=1}^{M} e_j$$

- Description:
  [會再翻成英文]
  敵人會專注於阻擋玩家繼續前進，迫使玩家與其發生衝突。MP 為加總所有空間動線權重 mp_i；E 為加總所有敵人於空間之動線權重 (e_j)，倘敵人並未落在動線上，權重則為0。為了達到隨著動線上的敵人愈多，對此適應性函數的影響力愈低，則取以 MP 為底 E 的對數。



Example gameplay of "Block"

Nine fitness functions – 3rd Intercept

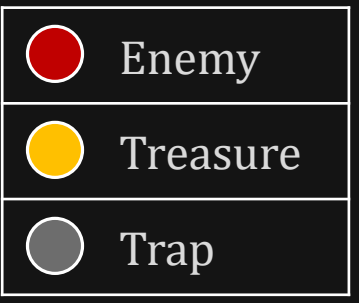
- Fitness function:

- Description:
  [會再翻成英文]
  與阻攔點近似，但敵人會被配置於動線附近非動線上，以快速追擊玩家為目的。各敵人 (E_i) 越接近空間動線各點 (MP_j) 時影響愈大，且動線權重 (mp_j) 亦會影響加權程度。

Example gameplay of "Intercept"

Nine fitness functions – 4th Patrol

- Fitness function:



- Description:
  [會再翻成英文]
  為確保各敵人擁有足夠的空間能夠進行移動。利用 $Cover_i$ 計算敵人 (E_i) 在指定半徑 (r) 內，能夠行動的座標點數量比例，$count(E_i, r)$ 代表指定半徑內敵人可以行動的座標點數量； $plane(E_i, r)$ 代表以敵人為中心的指定半徑內，平面上所有座標數量（包含不可通行的牆壁等類型）。將 $count(E_i, r)$ 和 $plane(E_i, r)$ 的比例作為可以行動的座標數量比例。另外，本次實驗為三維空間，因此有機會出現可行走的數量大於平面上所有座標數量，在此對兩者比較大小，取最大值作為所有座標數量，以確保 $Cover_i$ 的數值介於0至1之間。

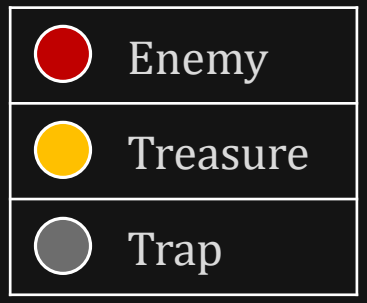
Example gameplay of "Patrol"

Nine fitness functions – 5th Guard

- Fitness function:

$$f_{grd} = \frac{1}{\|O\|} \times \sum_{i=1}^{\|O\|} \frac{\frac{\|E\|}{\|O\|} - \left| count(O_i, r) - \frac{\|E\|}{\|O\|} \right|}{\frac{\|E\|}{\|O\|}}, o_j \in O = \{Treasure, Exit\}$$

- Description:
  [會再翻成英文]
  為體現出敵人會保衛寶箱 (T)與出口 (E) 的現象，計算敵人 (E_i) 與關鍵性較高遊戲物件 (O_j) 之間的距離，倘若距離愈近則帶來的影響力愈大。

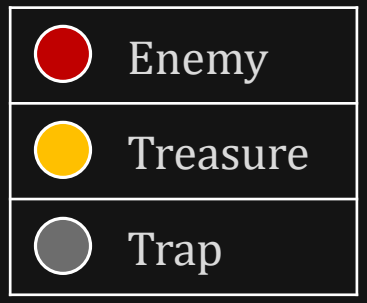


Example gameplay of "Guard"

Nine fitness functions – 6th Dominated

- Fitness function:

- Description:
  [會再翻成英文]
  當玩家可能所在動線上之位置 (MP_j) 與敵人的位置 (E_i) 具有高低差時，敵人便適合採取遠程攻擊；為了提供玩家思考對付遠程敵人的緩衝時間，將敵人配置於動線末端附近是較好的選擇，j 隨著動線的順序演進，影響程度逐漸增幅。

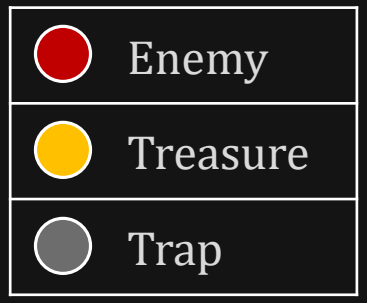


Example gameplay of "Dominated"

Nine fitness functions – 7th Support

- Fitness function:

- Description:
  [會再翻成英文]
  敵人 (E_i, E_j) 之間擁有一定程度的護援關係，當敵人彼此的距離愈低其影響程度越大，同時該敵人 (E_i) 必須遠離動線 (MP_k)。

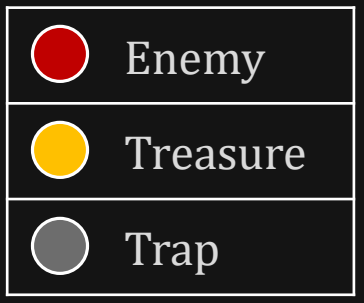


Example gameplay of "Support"

Nine fitness functions – 8th Cover

- Fitness function:

- Description:

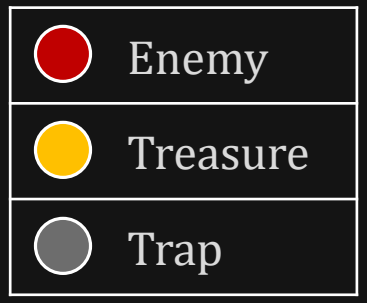


Example gameplay of "Cover"

Nine fitness functions – 9th Trap

- Fitness function:

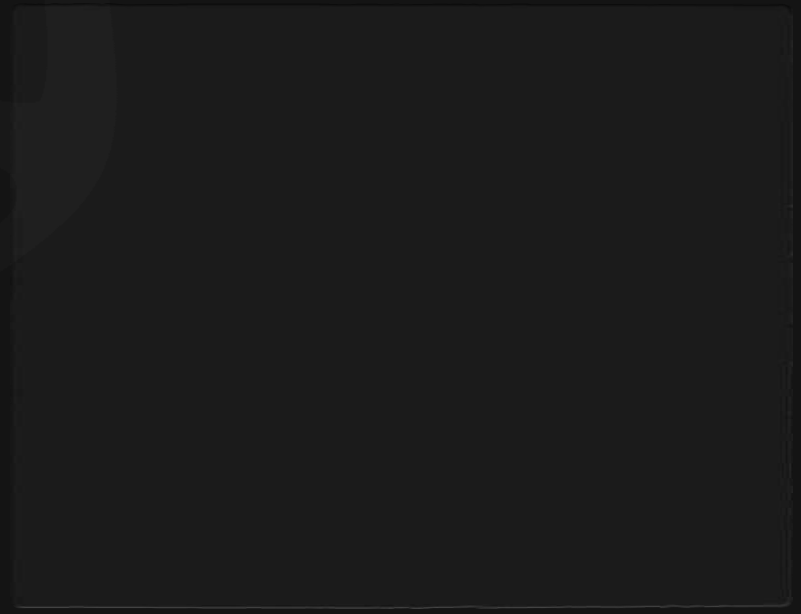
- Description:



Example gameplay of "Trap"

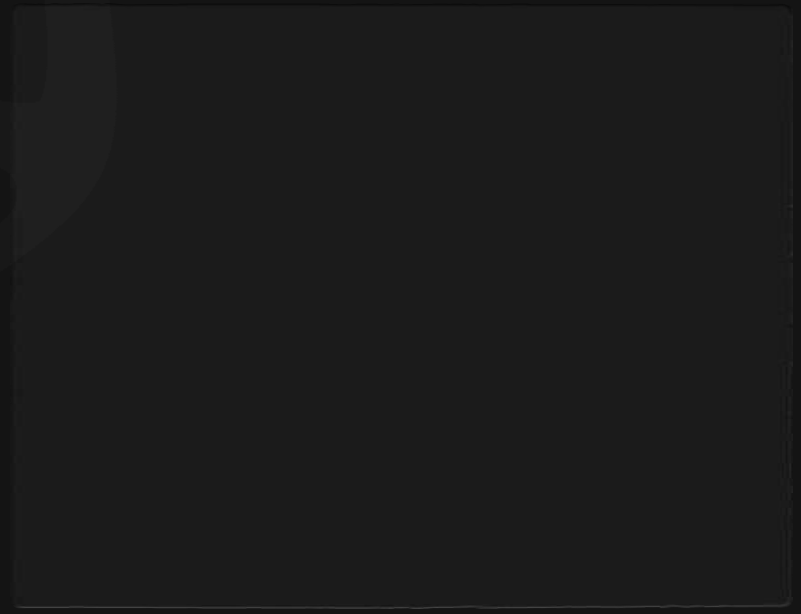
Usage flow

H Set the weights of fitness functions



Usage flow

I Export the completed level



Experimental Results

Conclusions and Contributions

Conclusions

Contributions

Future Work