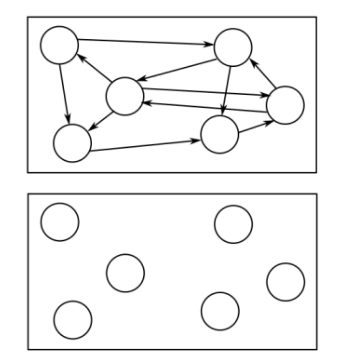
# LANDASAN TEORI

Pada bab ini akan dijelaskan mengenai teori yang dipakai untuk menyelesaikan masalah dalam pembuatan agent AI *Goal-Oriented Action Planning* di dalam *Turn-based Tactics Video Game*.

* 1. **Tinjauan Pustaka**

**2.1.1 Goal-Oriented Action Planning**

*Goal-Oriented Action Planning* adalah suatu metode *decision-making* yang dapat membuat suatu karakter tidak hanya melakukan apa yang akan dia lakukan, tetapi juga menentukan bagaimana cara ia melakukannya. Dengan struktur GOAP, GOAP mampu memfasilitasi suatu karakter dengan cara mempertahankan dan menggunakan ulang *behavior* tersebut disesuaikan dengan situasi dimana karakter tersebut berada. Sistem GOAP tidak akan mengganti kebutuhan akan Finite-state machine, tetapi lebih menyederhanakan FSM yang dibutuhkan (Orkin, 2003). GOAP merupakan teknik untuk *decision making* yang akan menghasilkan rantai aksi yang bernama *plan* untuk mencapai suatu *goal state* yang sudah ditentukan sebelumnya (Studiawan, Hariadi, & Sumpeno, 2018). GOAP biasanya memiliki 3 komponen utama dalam melakukan tugasnya, yaitu *goals* dan *actions*. *Goals* atau tujuan disini adalah sebuah kondisi yang harus dicapai oleh suatu *agent*, dengan melakukan *action*-*action* yang dapat mencapai kondisi tersebut.



Gambar 2.1 *Simplification* dari GOAP

Contoh dari pemakaian GOAP adalah ketika ada seorang agen yang ingin membuat sebuah kayu bakar. Untuk membuat sebuah kayu bakar, agen tersebut memiliki tiga tindakan yang sudah ditetapkan, yaitu :

1. *getAxe.* Biaya : 2. Prekondisi : “*An axe is available*”, “*doesn’t have an axe*”. Efek : “*has an axe*”
2. *chopLog*. Biaya : 4. Prekondisi : “*has an axe*”. Efek : “*make firewood*”
3. *collectBranches* Biaya : 8. Prekondisi : (tidak ada) Efek : “*make firewood*”

*World* state yang dimiliki agen tersebut adalah “*doesn’t have an axe*“ dan “*an axe is available*“. Berikut adalah GOAP *Planner* di dalam menyusun action sesuai *world state* yang ada

GOAL: "make firewood"

Current State: "doesn’t have an axe", "an axe is available"

Can action ChopLog run?

    NO - requires precondition "has an axe"

    Cannot use it now, try another action.

Can action GetAxe run?

    YES, preconditions "an axe is available" and "doesn’t have an axe" are true.

    PUSH action onto queue, update state with action’s effect

New State

    "has an axe"

    Remove state "an axe is available" because we just took one.

Can action ChopLog run?

    YES, precondition "has an axe" is true

    PUSH action onto queue, update state with action’s effect

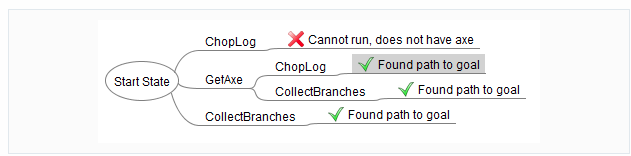
New State

    "has an axe", "makes firewood"

We have reached our GOAL of  "makes firewood"

Action sequence: GetAxe -> ChopLog

*Planner* tersebut akan berjalan melalui *action* yang lain juga, dan akan mencari *action* yang memiliki *lowest cost* yang mampu mencapai ke *goal* tersebut. *Tree* yang dibuat *planner* tersebut akan berbentuk seperti ini :



Gambar 2.2 *Tree* yang dibuat GOAP *Planner*

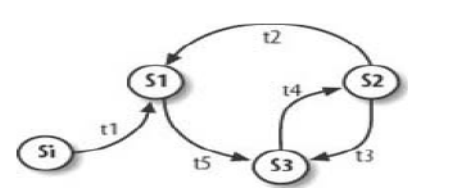
Melalui *planner* tersebut kita dapat melihat bahwa ada tiga rangkaian *action* yang dapat dijalankan, yaitu :

1. *getAxe -> chopLog* (membutuhkan biaya 6)
2. *getAxe -> collectBranches* (membutuhkan biaya 10)
3. *collectBranches* (membutuhkan biaya 8)

Dengan ini, *planner* akan mengambil *action* yang memiliki *cost* yang paling rendah, yaitu *action* pertama (*getAxe -> chopLog)* (Owens, 2014).

* + 1. **Finite State Machine**

*Finite state machine* adalah sebuah metodologi perancangan sistem control yang menggambarkan tingkah laku atau prinsip kerja sistem dengan menggunakan tiga hal berikut: *state, event,* dan *action* (Rahadian, Suyatno, & Maharani, 2016). Sistem dapat berpindah dari satu *state* ke *state* lain apabila suatu *event* terjadi. Biasanya transisi antar *state* disertai dengan aksi yang bisa dilakukan, baik aksi yang sederhana maupun kompleks.



Gambar 7.3 Struktur *Finite State Machine*

Pada gambar 7.3 terdapat 4 *state* {Si, S1, S2, S3} yang mungkin terjadi, setiap *state*-nya dapat berpindah *state* jika kondisi terpenuhi. Sebagai contoh *state* S1 dapat berpindah jika kondisi t5 terpenuhi (Rostianingsih, Budhi, & Wijaya, 2013).

* + 1. **Turn-Based Strategy Games (TBS)**

*Game* yang memiliki *genre turn-based strategy* (TBS) mendorong pemain untuk mengambil keputusan mengenai *action* jenis apa yang perlu dilakukan oleh pion-pion atau *unit-unit* yang dimiliki (Chandra, 2018). Turn-based Strategy Games bersifat putaran, dimana tiap pemain memiliki waktunya sendiri untuk menggerakkan *unit-unit* yang dimiliki oleh pemain tersebut. Tiap pemain memiliki objektif, yaitu mengalahkan musuh dengan *unit-unit* yang mereka miliki.



Gambar 2.3 *Advance Wars: Days of Ruin*

Sumber : Chandra (2018)

* 1. **Tinjauan Studi**

**2.2.1 *Applying Goal-Oriented Action Planning to Games***(Orkin, 2003)

* Masalah yang diangkat pada penelitian tersebut adalah peneliti ingin menganalisa cara kerja agent AI di sebuah *First-person shooter game* bernama *No One Lives Forever 2: A Spy in H.A.R.M.’s Way*, yang menggunakan metode *Goal-Oriented Action Planning.*
* Hasil penelitian menunjukkan bahwa karakter di dalam NOLF2memiliki *goal* sebanyak 25. Pada suatu waktu, satu *goal* akan aktif, mengontrol tingkah laku karakter. *Goal* di dalam NOLF2 jatuh ke dalam tiga kategori, yaitu *relaxed goals*, *investigative goals*, dan *aggressive goals*. *Relaxed goals* menyangkut *Sleep, work,* dan *patrol*. *Investigative Goals* menyangkut *suspicious investigate* dan *search*. *Aggressive Goals* menyangkut *combat situations*, seperti *chase, charge*, dan *attack from cover*. Meski secara konsep hamper sama, tetapi tujuan di *NOLF2* dan GOAP pada umumnya memiliki satu perbedaan. NOLF2 memiliki *embedded plan*. Di dalam penelitiannya, *search* yang digunakan adalah A\* *algorithm*, yang membutuhkan jumlah *cost* dari *node*, dan *heuristic distance* dari sebuah *node* ke sebuah *goal*.
* Perbedaan penelitian yang bersangkutan dengan penelitian ini terletak pada cara meneliti dan dataset yang diteliti. Pada penelitian yang bersangkutan, peneliti hanya menganalisa agent AI yang ada di dalam *game* NOLF2, dan dataset yang digunakan adalah dataset *first-person shooter game*.

**2.2.2 *Adaptive Goal Oriented Action Planning for RTS Games***(Magnusson & Hall, 2010)

* Masalah yang diangkat pada penelitian tersebut adalah peneliti ingin mengimplementasi *adaptive goal-oriented* pada AI di dalam sebuah *Real-time strategy game* karena peneliti melihat bahwa AI di dalam *game* RTS itu memiliki “fitur” lebih dan bergerak secara statis melalui *state-machines* dan semacamnya.
* Metode yang digunakan pada penelitian ini adalah metode simple *goal-oriented* *system*, dengan *adaptive behavior*, yang berarti metode tersebut akan membuat sebuah task yang akan *counter* perilaku musuh.
* AI Ice adalah sebuah AI yang dibuat di dalam penelitian ini. AI Ice membuat rencana-rencana, yang secara praktek sama dengan membuat variasi tugas yang mengarahkan kemenangan kepada dirinya. Untuk melakukan hal tersebut, AI Ice harus mendapatkan informasi di dalam *game* untuk menentukan, membuat, dan menjalankan *tasks* dengan *highest priority*. AI Ice akan selalu membuat *units* berdasarkan tipe senjata dan armor *units* musuh. AI Ice dibuat menggunakan Spring Engine
* Hasil penelitian menunjukkan bahwa adaptive AI yang dibuat oleh peneliti, yang memiliki “fitur” yang sama dengan *human player*, dapat mengalahkan *static* AI yang memiliki “fitur” lebih. Tetapi, prioritas *tasks* harus lebih diperhatikan, dan bisa dibilang akan menjadi susah untuk membuat sebuah prioritas yang bekerja pada “semua” situasi.
* Perbedaan penelitian yang bersangkutan dengan penelitian ini terletak pada dataset yang diteliti. Pada penelitian yang bersangkutan, peneliti menggunakan AI buatannya untuk diuji coba dengan AI yang ada di dalam *game* RTS tersebut, tidak ada pengujian dengan *human player*. Genre *game* yang diteliti juga berbeda.

**2.2.3 *Tactical Planning in Space Game using Goal-Oriented Action Planning***(Studiawan, Hariadi, & Sumpeno, 2018)

* Masalah yang diangkat pada penelitian tersebut adalah peneliti ingin membuat sebuah AI di dalam sebuah *game* bergenre *real-time tactics*. Peneliti ingin mencari tahu apakah AI menggunakan *Goal-Oriented Action Planning* dapat digunakan pada *space tactical game*.
* Metode yang digunakan pada penelitian ini adalah metode *Goal-Oriented Action Planning* untuk mengatur perilaku AI yang akan dibuat, berdasarkan *tujuan* yang ingin dicapaidan *state* AI tersebut berada. Penelitian ini juga menggunakan *Blackboard system* untuk memanage *data communication* berdasarkan *blackboard architecture model* dimana *general knowledge base* yang bernama *blackboard* digunakan dan diperbaharui oleh komponen-komponen di dalam sistem. Dengan menggunakan *blackboard system*, *module* di dalam *game* dapat lebih fokus ke fungsi daripada fokus untuk berkomunikasi dengan *module* lainnya.
* Hasil penelitian menunjukkan bahwa di dalam pembuatannya, GOAP membantu mengurangi kompleksitas ketika men-*design* AI. Menggunakan definisi yang ada di dalam penelitian tersebut, komposisi yang dinamis dapat terlihat. Di sisi lain, AI *designer* harus mengevaluasi semua aksi yang berhubungan dengan parameter ketika ada pergantian jenis parameter di dalamnya, dan hal tersebut diakui sebagai salah satu kesulitan menggunakan GOAP.
* Perbedaan penelitian yang bersangkutan dengan penelitian ini terletak pada tujuan penelitian dan dataset yang diteliti. Pada penelitian yang bersangkutan, tujuan penelitian yang dilakukan adalah bagaimana sebuah AI melakukan tugasnya ketika dihadapkan dalam berbagai kasus dan berbagai komponen.