Aplikasi Graf dalam Susunan Kombinasi Ninja dan Penempatannya pada Graf Deploy dalam Game Ninja Heroes

Razzan Daksana Yoni - 13521087
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521087@mahasiswa.itb.ac.id

***Abstract*—Ninja Heroes NewEra adalah salah satu *mobile online RPG* di *android*, permainan ini bergenre *adventure*, pemain harus mempersiapkan *hero* yang digunakan dan juga ada variabel lain untuk memperkuat *hero* tersebut. Salah satunya adalah memberikan *deploy* yang memiliki *stat* maksimal dari *hero-hero* pada deploy tersebut. Pemain perlu mencocokkan warna-warna yang dimiliki pada tiap hero untuk mendapatkan stat tambahan. Untuk itu, pemain dapat mengaplikasikan teori graf yaitu jumlah *stat* dari kombinasi-kombinasi *vertex* yang dibentuk untuk mempermudah pemain dalam menentukan kombinasi dan penempatan *hero*.**

***Keywords*—Ninja Heroes NewEra, *deploy*, graf, Adjacency matrix**

I. PENDAHULUAN

Ninja Heroes NewEra merupakan permainan *online RPG* yang dikembangkan oleh Kageherostudio dan dirilis pada Agustus 2022. Pemain harus menyusun strategi dan *hero* yang akan digunakan, seperti *chakra*, ninjutsu *skill*, *deploy*. Pemain harus menentukan kombinasi apa yang paling cocok untuk melawan musuh. Pada makalah ini, hanya difokuskan untuk mencari kombinasi dan penempatan *hero* pada graf *deploy* yang terbaik berdasarkan *hero* yang dimiliki.

Untuk menentukan *deploy* pemain perlu menyiapkan setidaknya 15 *hero* untuk memaksimalkan *stat*. Setiap kombinasi *hero* biasanya punya karakteristik unik yaitu menambah *stat* ataupun menjadi *trigger attack* yang dapat membantu *hero* utama ketika melawan musuh. Biasanya, kombinasi tersebut dilihat dari keadaan pada *anime Naruto*.

Gambar 1. Tampilan Laman Utama Ninja Heroes NewEra
(Sumber : Arsip Penulis)

Stat pada kombinasi *hero* bisa saja berbeda lebih banyak ataupun lebih sedikit. Dan *stat* juga memiliki karakteristik unik yaitu *HP*, *attack*, *defense*, dan *agility*. Hal ini pun bisa menjadi prioritas pemain dalam menentukan *stat* apa yang cocok untuk digunakan. Banyaknya *hero* yang ada pada gim ini tentu membuat pemain harus cerdas dalam menentukan strategi *hero* yang dapat memaksimalkan *stat.* Pemilihan Peletakan *hero* pun dapat berpengaruh pada keberhasilan permainan. Pada proses pemilihan *hero*, graf dapat membantu pemain agar lebih mudah dalam memilih kombinasi *hero*.[1]

II. LANDASAN TEORI

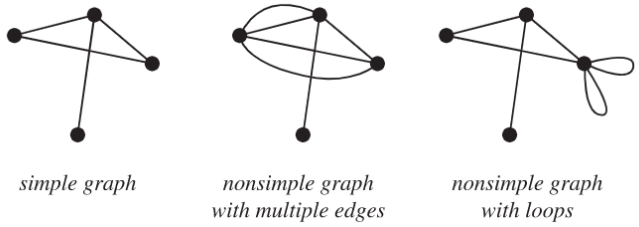
A. *Teori Graf*

Graf merupakan sekumpulan titik dan garis yang digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut. Titik-titik pada graf disebut simpul dan garis-garis yang menjadi penghubung antara simpul adalah sisi. Sisi ganda pada graf adalah ketika terdapat lebih dari satu sisi yang menghubungkan dua simpul yang sama dan sisi gelang pada graf adalah ketika sisi yang menghubungkan satu simpul yang sama.

B. *Jenis-jenis Graf*

Berdasarkan ada dan tidaknya sisi gelang atau sisi ganda, graf digolongkan menjadi dua jenis [2], yaitu

1. Graf sederhana (*simple graph*).
   Graf sederhana adalah graf yang tidak mengandung sisi ganda maupun sisi gelang.
2. Graf tak-sederhana (*unsimple-graph*).
   Graf tak-sederhana adalah graf yang mengandung sisi ganda atau sisi gelang. Graf tak-sederhana dibagi menjadi dua, yaitu
   a. Graf ganda (*multi-graph*)
      Graf dapat dikatakan sebagai graf ganda ketika graf tersebut mengandung sisi ganda.
   b. Graf semu (*pseudo-graph*)
      Graf dapat dikatakan sebagai graf semu ketika graf tersebut mengandung sisi gelang.



Gambar 2.1 Contoh graf sederhana (kiri), graf ganda (tengah), dan graf semu (kanan).
(Sumber : https://mathworld.wolfram.com/SimpleGraph.html diakses pada 10 Desember 2022 pukul 05.15)

Berdasarkan orientasi arah pada sisi, graf dapat dibedakan menjadi dua jenis, yaitu

1. Graf tak-berarah (*undirected graph*)
   Graf tak berarah adalah graf yang tidak mempunyai orientasi arah pada sisi-sisinya.
2. Graf berarah (*directed graph* atau *digraph*)
   Graf berarah adalah graf yang setiap sisinya mempunyai orientasi arah.

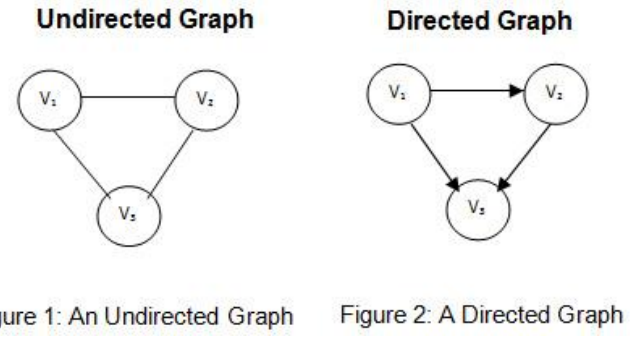


Figure 1: An Undirected Graph Figure 2: A Directed Graph

(Gambar 2.2 Contoh graf tak-berarah (kiri) dan graf tak berarah (kanan).
(Sumber : http://omtlab.com/directed-and-undirected-graph/ diakses pada 10 Desember 2022 pukul 05.20)

C. Terminologi Graf

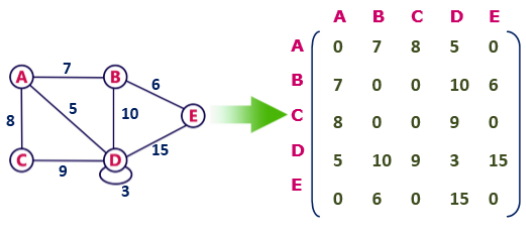
Terminologi atau istilah yang biasa digunakan pada teori graf, yaitu

1. Ketetanggaan (*adjacent*)
   Ketetanggaan terjadi ketika dua buah simpul pada graf saling bertetanggaan atau kedua simpul tersebut dihubungkan oleh kedua sisi.
2. Bersisian *(incidency*)
   Sisi graf yang bersisian adalah ketika dua buah simpul pada sisi tersebut dihubungkan oleh sisi tersebut.
3. Simpul terpencil (*isolated vertex*)
   Simpul dikatakan terpencil ketika simpul tersebut tidak mempunyai sisi yang bersisian dengannya.
4. Graf kosong (*null graph* atau *empty graph*)
   Graf kosong adalah graf yang himpunan sisinya adalah himpunan kosong atau graf yang di dalamnya tidak terdapat sisi sama sekali
5. Derajat (*degree*)
   Derajat dari suatu simpul menandakan jumlah sisi yang bersisian dengan simpul tersebut.
6. Lintasan (*path*)
   Lintasan merupakan kumpulan dua buah simpul yang menunjukkan lajur pada suatu graf, simpul pertama menunjukkan simpul asal, simpul kedua menunjukkan simpul tujuan. Lintasan biasa digunakan untuk menunjukkan Langkah-langkah dari suatu simpul ke simpul yang lain pada graf.
7. Siklus (*cycle*) atau Sirkuit (*circuit*)
   Siklus atau sirkuit merupakan lintasan pada suatu graf dengan simpul awal yang sama dengan simpul akhir yang berarti lintasan tersebut kembali ke simpul yang sama.
8. Keterhubungan (*connected*)
   Keterhubungan antar dua buah simpul terjadi ketika terdapat lintasan yang menghubungkan kedua simpul tersebut. Apabila semua simpul dalam graf tersebut saling terhubung, maka graf tersebut merupakan graf terhubung (*connected graph*). Sedangkan, apabila terdapat simpul yang tidak terhubung dengan simpul lain yang berada pada graf tersebut, maka graf tersebut merupakan graf tak-terhubung (*disconnected graph*).
9. Upagraf (*subgraph*) dan Komplemen Upagraf
   Suatu graf dikatakan upagraf jika graf tersebut merupakan subset dari suatu subgraph, sedangkan upagraf merupakan bagian pada graf yang bukan termasuk dari upagraf tersebut.
10. Upagraf merentang (*spanning subgraph*)
    Upagraf merentang adalah sebuah upagraf yang mengandung semua simpul pada graf.
11. *Cut-set*
    Cut-set adalah himpunan sisi pada suatu graf yang apabila sisi tersebut dihilangkan, graf tersebut akan berubah menjadi graf tak-terhubung.
12. Graf berbobot (*weighted graph*)
    Graf berbobot adalah graf yang pada sisi-sisinya mengandung sebuah harga (bobot).

D. Representasi Graf

Dalam merepresentasikan graf, dapat memilih struktur atau bentuk beragam. Hal ini, ditentukan dengan pemrosesan dalam algoritma yang diambil. Representasi graf pada makalah ini, yaitu

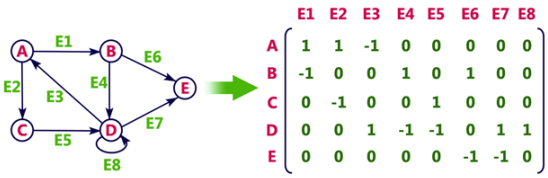
1. Matriks Ketetanggaan (*adjacency matrix*)
   Matriks ketetanggan dapat merepresentasikan sebuah graf yaitu dengan matriks berukuran n*vertex*Xn*vertex*. Setiap elemen pada matriks ini merepresentasikan keterhubungan *vertex* dengan *vertex* yang terhubung. Matriks ketetanggaan juga dapat merepresentasikan graf berarah maupun graf tak berarah.



(Gambar 2.3 Representasi graf dengan matriks ketetanggaan)

(Sumber: https://www.javatpoint.com/graph-theory-graph-representations#:~:text=In%20graph%20theory%2C%20a%20graph,to%20it%20by%20an%20edge).)

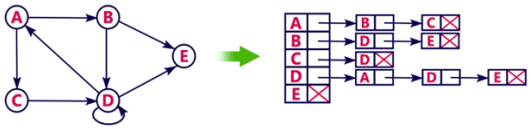
2. Matriks Bersisian (*incidency matrix*)
   Matriks Bersisian dapat merepresentasikan sebuah graf yaitu dengan matriks berukuran n*vertex*X*nedge*. Setiap elemen pada matriks ini merepresentasikan keterhubungan *vertex* dengan *edge* yang bersisian. Sama seperti matriks ketetanggaan, matriks bersisian juga dapat merepresentasikan graf berarah maupun tak berarah.



(Gambar 2.4 Representasi graf dengan matriks bersisian)

(Sumber: https://www.javatpoint.com/graph-theory-graph-representations#:~:text=In%20graph%20theory%2C%20a%20graph,to%20it%20by%20an%20edge).)

3. Senarai Ketetanggaan (*adjacency list*)
   Senarai ketetanggaan dapat merepresentasikan sebuah graf, biasanya digambarkan dengan *array* atau *linked list*. Elemen *head* pada *linked list* adalah *vertex* acuan dan *elemen* selain *head* adalah *vertex* yang bertetanggaan dengan *vertex* tersebut. Senarai ketetanggaan juga dapat merepresentasikan graf berarah maupun tidak berarah.



(Gambar 2.5 Representasi graf dengan senarai ketetanggaan)

(Sumber: https://www.javatpoint.com/graph-theory-graph-representations#:~:text=In%20graph%20theory%2C%20a%20graph,to%20it%20by%20an%20edge).)

III. Aplikasi Graf Untuk Menentukan Hero dan Penempatan Hero Pada Deploy

Variasi penempatan hero pada deploy yang dapat dibentuk di dalam gim Ninja Heroes adalah bermacam-macam atau jika dikalkulasikan kombinasi tersebut berjumlah $12! * 3!$, hal ini dikarenakan banyaknya kemungkinan kombinasi letak penentuan *hero-hero* tersebut. Prioritas *stat* juga dapat mempengaruhi hasil kombinasi penempatan *deploy* hal ini menyebabkan pemain perlu mengkalkulasikan penempatan mana yang paling sesuai dari yang pemain inginkan. Graf pada deploy pada dasarnya ditentukan oleh hero utama karena *hero* utama pada *deploy* tidak dapat diubah pada graf *deploy*.

(Gambar 3.1 Hero Utama pada gim *Ninja Heroes*)
(Sumber : Arsip Penulis)

Tiap *hero* juga memiliki karakteristik yang unik yaitu *stat* pada bagian atas, bawah, kanan, dan kiri. Warna-warna tersebut pula mempunyai makna tersendiri, warna hijau melambangkan *stat HP*, warna merah melambangkan *stat attack*, warna biru melambangkan *stat defense*, dan warna kuning melambangkan stat *agility*. Tiap *hero* biasanya memiliki kombinasi warna tersebut berbeda-beda.

(Gambar 3.2 Contoh karakteristik warna-warna pada hero)
(Sumber : Arsip Penulis)

Pemain dapat menambahkan *stat* dengan mencocokkan warna-warna tersebut dengan *hero* yang memiliki warna yang sama pada graf deploy, seperti jika warna merah ada di bawah pemain perlu mencocokkan pula warna pada *hero* yang barisnya tepat di bawah dan pada kolom yang sama pada *hero* tersebut dengan warna di atas yang sama, untuk memperoleh *stat* tambahan tersebut. Pemain hanya dapat memasukkan jumlah simpul atau *hero* pada graf *deploy* berjumlah lima belas yaitu tiga *hero* utama dan dua belas *hero* lain selain *hero* utama. Juga, pemain hanya bisa memasukkan *hero* yang sama satu kali, jadi tidak aka nada *hero* yang muncul lebih dari satu kali pada graf *deploy*. Oleh karena itu, pemain perlu menentukan kondisi yang cocok untuk menghadapi permasalahan tersebut.

(Gambar 3.3 Contoh deploy pada gim Ninja Heroes)
(Sumber : Arsip Penulis)

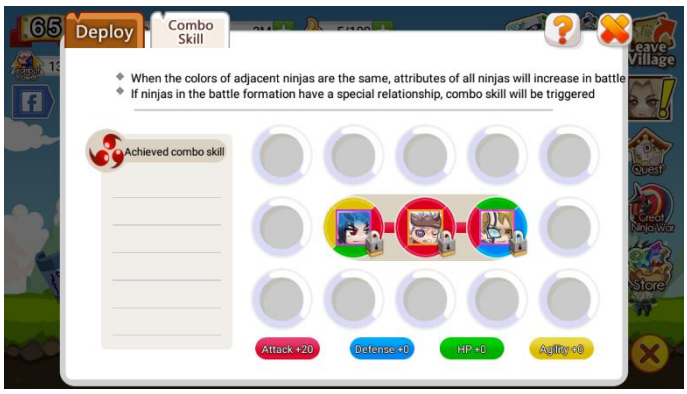
Pemain dapat menetukan kombinasi penempatan *hero-hero* tersebut dengan *stat* maksimum adalah dengan mencari jumlah derajat terbesar atau jumlah sisi terbanyak yang mungkin pada graf *deploy* tersebut. Dari kasus ini, menunjukkan bahwa ada kemungkinan *stat* lebih dari satu dengan jumlah sisi atau jumlah derajat maksimum. Karenanya, pemain perlu menentukan prioritas *stat*-nya dalam penentuan penempatan graf. Jika ingin mendapatkan *stat HP* maksimum maka pemain perlu mencari sisi yang berwarna hijau juga maksimum, jika ingin mendapatkan *stat attack* maksimum maka pemain perlu mencari sisi yang berwarna merah juga maksimum, jika ingin mendapatkan *stat defense* maksimum maka pemain perlu mencari sisi yang berwarna biru juga maksimum, dan jika ingin mendapatkan *stat agility* maksimum maka pemain perlu mencari sisi yang berwarna kuning juga maksimum. Namun, hal ini juga berpengaruh terhadap *hero-hero* yang kita taruh di dalam deploy.

Proses untuk mendapatkan deploy bukan suatu hal yang mudah, dikarenakan kemungkinan kombinasi yang banyak dan prioritas yang dapat menyulitkan pemain dalam penentuan peletakan *hero,* dan juga dimungkinkan pemain tidak memiliki *hero* yang pas dalam memasangkan kecocokan warna atau mungkin *deploy* dari kombinasi *hero* tidak bagus jika hany mementingkan graf *deploy*. Proses ini membutuhkan waktu yang cukup lama jika kita tidak tahu caranya untuk mendapatkan *stat* maksimum tersebut, sebaliknya proses ini akan sangat cepat jika kita mengerti algoritma dalam peletakan *hero-hero* tersebut.

Implementasi dan struktur data dari pemilihan dan penempatan *hero* dapat menggunakan konsep graf, terkhusus, graf berbobot dengan representasi matriks ketetanggaan, dan kebersisian. Pada penentuan pemilihan *hero* yang akan digunakan pada graf *deploy* pemain perlu menentukan *hero* utamanya terlebih dahulu, karena *hero* utama akan menentukan *combo skills* dan graf *deploy.* Pemain dapat memilih tiga *hero* utama berdasarkan *hero* yang pemain miliki. Setelah menentukan *hero* utama, pemain juga perlu melihat keterkaitan antara *hero-hero* tersebut, seperti pada Gambar 3.3, ketika *hero-hero* pada graf *deploy* adalah subset dari suatu himpunan *combo skills* maka *stat* juga akan meningkat, untuk menentukan *hero-hero* tersebut pemain perlu melihat *hero* yang dimiliki, lalu cari *hero* yang mempunyai keberirisan pada tiap himpunan *combo skills* terbanyak dan dikalkulasikan pula dengan jumlah *stat* hasil dari *combo skills* tersebut, setelah itu lihat pada himpunan *hero* untuk menentukan bobot yang akan didapatkan dan juga tentukan prioritas *stat* yang akan diambil, untuk itu pemain dapat menggunakan graf berarah dengan implementasi matriks ketetanggaan dengan ukuran 15x15 dalam menetukan *hero-hero* tersebut dan memberikan bobot untuk tiap himpunan yang terbentuk. Lalu, setiap himpunan tersebut bagi *stat* hasil dari *combo skills* dengan sejumlah *hero* yang ada pada himpunan tersebut, lalu masukkan *stat* tersebut ke dalam matriks ketetanggaan dengan predecessornya adalah *hero* yang paling kiri dalam himpunan dan suksesornya adalah *hero* lainnya. Lalu lakukan terus hingga jumlah *hero* yang terdapat pada graf *deploy* adalah dua belas.

Setelah pemain menentukan kombinasi *hero* pada *deploy*, pemain perlu menentukan peletakan *hero-hero* tersebut. Untuk itu pemain terlebih dahulu mengkarakteristikkan warna-warna yang dimiliki oleh tiap *hero*. Setelah itu, pemain dapat menaruh *hero* tersebut ke dalam *array of heroes* (tidak ada di dalam gim perlu diimplementasikan) untuk memudahkan pemain mengingat *hero* yang telah termasuk di dalam graf atau tidak.

Setelah itu, tentukan peletakan basis yaitu peletakan *hero* utama pada graf (prioritas *stat* dapat mempengaruhi hasil), jika pemain tidak memikirkan *stat* yang menjadi prioritas hanya memikirkan *stat* maksimum yang dapat diperoleh taruh *hero* utama tersebut dengan ketetanggaan maksimum atau jika tidak terdapat ketetanggaan yang mungkin terjadi maka taruh *hero* dengan *stat* yang diinginkan di sebelah kanan atau kiri karena ini akan memungkinkan ketetanggaan lebih banyak pada graf *deploy* tersebut, seperti jika *hero* diletakkan di sebelah kiri atau kanan *hero* tersebut akan memiliki tiga kemungkinan ketetanggaannya yaitu atas bawah dan kiri untuk *hero* yang diletakkan di sebelah kiri atau kanan untuk *hero* yang diletakkan di sebelah kanan, sebaliknya jika *hero* tersebut ditaruh di tengah hanya memiliki dua kemungkinan yaitu ketetanggaan dengan yang di atas dan di bawah, agar jumlah sisi dari graf bernilai maksimum.



(Gambar 3.4 Contoh penempatan kombinasi *hero* utama)
(Sumber : Arsip penulis)

Setelah langkah sebelumnya, pemain harus meletakkan sisa dua belas *hero* ke dalam graf *deploy*. Untuk itu, penulis baru terpikirkan algoritma yang mungkin untuk mendapatkan *stat* maksimum. Cara pertama adalah taruh sisa hero yang cocok dengan *stat* atas atau bawah dari *hero* utama sebelah kiri maupun kanan agar dapat memudahkan pemain dalam menentukan hasil akhir dari graf *deploy*, jika tidak ada yang cocok dengan warna-warna yang dihasilkan maka pemain dapat menaruh *hero* juga pada *hero* utama yang terletak di tengah, masukkan *hero* tersebut ke dalam *array* pertama atas dan *array* pertama bawah. Setelah itu, dengan cara yang sama lakukan juga untuk bagian sebelah kanan maupun sebelah kiri dari *hero* utama, lalu *hero* tersebut masukkan ke dalam *array* kedua kiri dan *array* kedua kanan. Apabila terdapat keteririsan antara *array* pertama dan ke dua, jika ada *array* yang hanya memiliki satu *hero* maka hapus *hero* pada *array* lain karena tidak mungkin dalam suatu saat yang bersamaan *hero* digunakan lebih dari satu kali, dan ini akan memungkinkan graf memiliki jumlah sisi maksimum.

Setelah melakukan langkah-langkah sebelumnya, tersisa *heroes* yang belum masuk ke dalam *array* manapun. Maka dari itu, masukkan sisa *heroes* tersebut ke dalam *array* sisa *heroes.* Setelah itu, satu per-satu pasangkan *hero* ke dalam graf *deploy*

yaitu *heroes* yang ada pada *array* pertama maupun *array* kedua secara permutasi. Lalu, lakukan juga kepada sisa *heroes* yaitu mencocokkan dengan warna-warna yang telah dimasukkan ke dalam graf *deploy* secara permutasi. Setiap kombinasi, hitung jumlah sisi yang terbentuk, ketika graf sudah mencapai jumlah sisi maksimum dengan semua kemungkinan yang ada, pemain dapat memprioritaskan *stat* yang akan menjadi prioritas, karena kombinasi penempatan mungkin menghasilkan jumlah sisi yang sama, tetapi mempunyai perolehan *stat* yang berbeda . Setelah melakukan langkah-langkah di atas, pemain dapat mendapatkan kombinasi dan penempatan *heroes* dengan *stat* paling maksimum juga prioritas yang diinginkan.

IV. IMPLEMENTASI

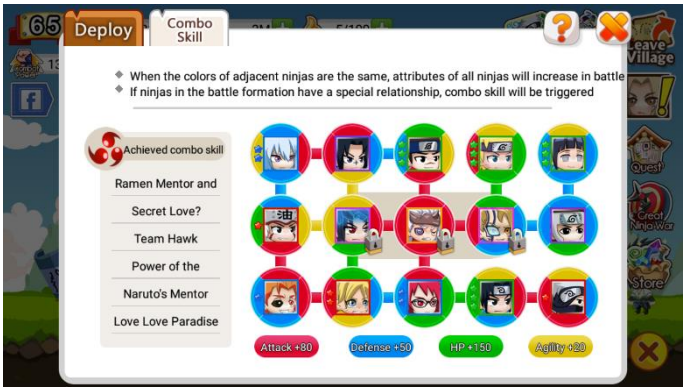
Setelah melihat bagian tiga, sebenarnya ada banyak sekali faktor yang mempengaruhi bentuk dan kombinasi *hero*, salah satunya adalah kepemilikan *hero*. Jadi, untuk itu pemain dapat menyadur dengan implementasi yang penulis buat. Di langkah awal pemain diberi petunjuk untuk menentukan *hero* utama terlebih dahulu, berdasarkan kepemilikan, *stat*, dan tipe *chakra* yang terdapat pada *hero* yang dimiliki penulis maka penulis memilih seperti yang ada pada gambar 3.1 yaitu *Rinnegan Sasuke*, *Six Paths Madara*, dan *Boruto Karma*. Setelah itu pemain diberi petunjuk untuk menentukan prioritas dan prioritas *stat* yang diinginkan penulis adalah *HP*, maka dari itu setelah mencari subset dan kecocokan pada tiap-tiap *combo skills* yang dihasilkan penulis mendapatkan *hero-hero* yaitu *Jugo*, *Boruto*, *Iruka*, *Suigetsu*, *Kakashi*, *Jiraiya*, *Karin*, *Naruto*, *Sasuke*, *Cursed Seal Sasuke*, *Yamato*, *Hinata*, karena setelah dikalkulasikan dengan cara yang telah diberikan kombinasi *hero* tersebut memiliki *stat HP* maksimum. Matriks ketetanggaan yang dihasilkan adalah

Tabel 4.1 Matriks Ketetanggaan *stat* dari kombinasi *Heroes*

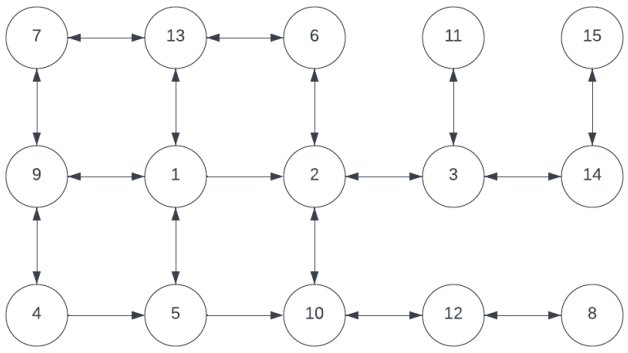
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	6	0	0	0	6	0	6	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	5	0	0	0	0	0	5	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	5	0	0	0	0	0	0	0	0	4
12	0	0	0	2	0	0	2	0	0	1	0	0	0	0	0
13	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0

Dari matriks di atas terhitung jumlah *stat* yang dihasilkan adalah sejumlah 56. Setelah itu, penulis menentukan kondisi *hero* utama seperti yang ada pada Gambar 3.4 karena kondisi tersebut memiliki *stat* yang maksimum yang mungkin dicapai. Setelah itu, diberi petunjuk untuk menentukan *array* pertama atas dan bawah. *Heroes* yang dapat memungkinkan pada *array* pertama atas adalah *Naruto* dan *Karin* untuk kecocokan warna hijau, *Boruto* dan *Cursed Seal Sasuke* untuk kecocokan warna kuning dan untuk *array* pertama bawah adalah *Boruto* untuk kecocokan warna hijau dan *Karin* untuk kecocokan warna merah. Dikarenakan ada keteririsan antara dua *array* tersebut maka pemain perlu menghapus salah satunya yaitu yang memiliki kondisi *hero* lebih dari satu yaitu pada *array* pertama atas. Lalu, pemain perlu menentukan *hero* yang ada pada bagian kanan dan kiri untuk bagian kanan yaitu kecocokan dengan warna kuning *hero* yang memungkinkan adalah *Jiraiya* dan *Iruka* dan untuk bagian kiri yaitu kecocokan dengan warna biru adalah *Hinata* dan *Yamato*. Dapat dilihat bahwa tidak ada keteririsan antara *array* yang telah terbentuk. Setelah itu kombinasikan tiap *hero* pada *array* yang telah didapatkan secara permutasi, lalu lakukan penempatan *hero* sisa dengan ketercocokan warna yang sama.

Setelah melakukan hal di atas penulis mendapatkan jumlah sisi maksimum yang mungkin di dapatkan oleh graf deploy adalah sejumlah delapan belas. Dan untuk prioritas *stat* pada graf *deploy* yang diinginkan oleh penulis adalah *stat* dengan *attack* maksimum. Dan dari itu didapatkan graf *deploy* sebagai berikut



(Gambar 3.5 Graf deploy hasil kecocokan dari algoritma)
(Sumber : Arsip Penulis)



(Gambar 3.6 Hasil graf *deploy*)
(Sumber : Arsip Penulis)

Dapat dilihat hasil dari algoritma yang telah diberikan menghasilkan graf yang jumlah sisinya adalah delapan belas, yaitu jumlah sisi maksimum yang dapat dibentuk.

V. KESIMPULAN

Representasi struktur data graf dimanfaatkan dalam banyak kasus..Kasus yang ada pada makalah ini yaitu menentukan susunan kombinasi *hero* dan peletakannya pada *graf* dapat dipermudah dengan menggunakan konsep graf, meskipun ada banyak cara untuk menyelesaikan masalah tersebut, seperti teori himpunan, relasi, kombinatorial, dan pohon. Dengan implementasi ini kita dapat mendapatkan *stat* maksimum yang mungkin diperoleh seperti pada Gambar 3.6 yang menghasilkan graf dengan jumlah sisi maksimum

VII. UCAPAN TERIMA KASIH

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas berkat dan karunia-Nya, penulis dapat menyelesaikan makalah yang berjudul “Aplikasi Graf dalam Susunan Kombinasi Ninja dan Penempatannya pada Graf Deploy dalam Game Ninja Heroes” dengan lancar. Penulis mengucapkan terimakasih sebesar-besarnya kepada ibu Dr. Nur Ulfa Maulidevi, bapak Dr. Rinaldi Munir, dan ibu Dr. Fariska Zakhralativa, sebagai pengampu dalam menjalani kuliah IF2120 Matematika Diskrit 2022/2023, atas arahan, bimbingan dan ilmu yang telah diajarkan kepada penulis dan teman-teman. Penulis juga mengucapkan terimakasih kepada orangtua, saudara, dan teman-teman yang selalu memberi dukungan sehingga penulis dapat menyelesaikan makalah ini dengan lancer Penulis juga meminta maaf jika ada kekeliruan atau kesalahan yang terdapat pada makalah ini, baik sengaja maupun tidak disengaja. Penulis sangat terbuka terhadap masukan, kritikan, dan saran terhadap makalah ini. Akhir kata, penulis berharap semoga makalah ini dapat bermanfaat.

REFERENSI

[1] https://gamefinity.id/review/ninja-heroes-new-era-turn-based-yang-bangkit-dari-kubur/
[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf
[3] https://mathworld.wolfram.com/SimpleGraph.html
[4] https://www.geeksforgeeks.org/graph-and-its-representations/
[5] http://omtlab.com/directed-and-undirected-graph/
[6] https://www.javatpoint.com/graph-theory-graph-representations#:~:text=In%20graph%20theory%2C%20a%20graph,to%20it%20by%20an%20edge

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 11 Desember 2022

Ttd

Razzan Daksana Yoni 13521087