BAB 7

KESIMPULAN DAN SARAN

7.1 Kesimpulan

Berikut diberikan tujuan penelitian yang telah tercapai berdasarkan tujuan penelitian yang diberikan pada Sub-bab 1.3:

- **Memodelkan papan *Light Up* sehingga dapat digunakan dalam proses pencarian solusi.** Pada skripsi ini papan *Light Up* dimodelkan dalam bentuk objek `Board`, di mana setiap petak dimodelkan dalam bentuk objek `Cell`. Papan pada objek `Board` dimodelkan dalam bentuk *np.array* dua dimensi yang memiliki elemen berupa objek `Cell`. Objek `Cell` memiliki beberapa atribut yang digunakan dalam proses pencarian solusi seperti `type`, `available`, dan `countAvailable`. Atribut `type` merepresentasikan jenis petak, atribut `available` merepresentasikan kondisi sekeliling petak, dan atribut `countAvailable` merepresentasikan jumlah petak putih yang dapat diletakkan lampu di sekeliling petak hitam berangka 1–4. Peletakkan lampu pada papan diimplementasikan pada objek `Board`.
- **Mengimplementasikan Algoritma GWO dalam proses pencarian solusi papan *Light Up*.** Pada skripsi ini Algoritma GWO diimplementasikan dalam dua tahap. Pada tahap pertama, Algoritma GWO digunakan untuk mencari posisi peletakkan lampu di sekeliling petak hitam berangka 1–4 yang dapat diletakkan lampu di sekelilingnya. Kandidat solusi tahap pertama dipilih menggunakan Algoritma *Roulette Wheel* yang telah dimodifikasi. *Pseudocode* modifikasi Algoritma *Roulette Wheel* dijelaskan pada Sub-bab 4.6. Tahap pertama menghasilkan kumpulan kandidat solusi berupa kumpulan kandidat solusi hasil pemilihan dan ditambahkan dengan serigala yang memiliki nilai *fitness* terbaik dari hasil pencarian dengan Algoritma GWO. Terkait pemilihan kandidat dijelaskan pada Sub-sub-bab 3.5.5. Pada tahap kedua, Algoritma GWO digunakan untuk mencari posisi peletakkan lampu pada petak putih yang belum tersinari oleh lampu dan dapat diletakkan oleh lampu di setiap papan kandidat hasil tahap pertama. Penjelasan terkait rancangan implementasi Algoritma GWO pada setiap tahap dijelaskan pada Sub-bab 3.5.
- **Merancang fungsi evaluasi yang dapat mengkuantifikasi kondisi papan saat ini.** Fungsi evaluasi yang dirancang pada skripsi ini memanfaatkan aturan dari permainan *Light Up* dan heuristik yang ditemukan pada permainan *Light Up*. Fungsi evaluasi dirancang dengan memberikan hukuman berupa penambahan nilai *fitness* pada serigala yang melakukan pelanggaran aturan dan heuristik, sehingga pada skripsi ini proses pencarian solusi dilakukan dengan meminimalkan nilai *fitness*. Terkait fungsi evaluasi yang dirancang, dijelaskan pada Sub-sub-bab 3.5.4.
- **Menyalin papan secara otomatis dari situs penyedia papan dan mengubahnya menjadi *input* yang dapat diolah.** Pada skripsi ini, penyalinan papan secara otomatis dari situs penyedia papan dilakukan dengan *web scraping*. *Web scraping* dilakukan dengan menggunakan Selenium yang diimplementasikan dalam bahasa pemrograman Python. Papan yang telah diperoleh dengan menggunakan Selenium diolah sesuai dengan identifikasi petak yang digunakan, terkait hal ini dijelaskan pada Sub-bab 3.2. Kumpulan papan yang telah diolah sesuai dengan identifikasi petak yang digunakan disimpan pada sebuah *file* dengan

menggunakan modul Pickle. Pickle mampu menyimpan objek dalam sebuah *file*. Terkait pembuatan *dataset input* dijelaskan pada Sub-sub-bab 3.3.1.

- **Menggunakan Selenium dalam menyimulasikan model pada situs permainan *Light Up*.** Pada skripsi ini, Selenium dimanfaatkan dalam menyimulasikan model pada situs permainan *Light Up*. Pada situs permainan *Light Up* yang dapat diakses melalui https://www.puzzle-light-up.com/ pemain dapat meletakkan lampu pada sebuah petak putih dengan menekan tombol kiri *mouse*, dan pemain dapat memberikan tanda silang pada sebuah petak putih dengan menekan tombol kanan *mouse*. Interaksi seperti yang dilakukan oleh pemain dapat dilakukan secara otomatis dengan menggunakan Selenium. Selenium mendukung interaksi berupa menekan tombol kiri *mouse* dan menekan tombol kanan *mouse*, sehingga hasil pencarian solusi papan pada situs permainan dapat disimulasikan oleh model. Pada Sub-bab 6.4 diberikan contoh hasil simulasi model yang menggunakan Selenium pada situs permainan *Light Up* dengan jenis papan 25×25 *easy*, dan dapat diselesaikan dalam waktu 9.17 detik.

7.2 Saran

Berdasarkan eksperimen yang dilakukan pada Bab 6, model yang dirancang hanya mampu menyelesaikan jenis papan hingga 25×25 *easy* hanya jika *preprocessing* digunakan, serta model yang dirancang hanya mampu menyelesaikan jenis papan hingga 14×14 *easy* hanya jika *preprocessing* tidak digunakan. Berikut diberikan saran yang dapat dijadikan panduan apabila penelitian skripsi ini dilakukan lebih lanjut:

- Mencari heuristik lain pada permainan *Light Up* yang dapat memperkecil ruang pencarian pada tahap *preprocessing*. Pada skripsi ini, *preprocessing* dilakukan sesuai heuristik yang ditemukan. Berdasarkan kesimpulan eksperimen yang dijelaskan pada Sub-bab 6.3, penggunaan *preprocessing* menjadi faktor penting dalam menemukan solusi papan, sehingga apabila ditemukan heuristik lain yang dapat digunakan dalam melakukan *preprocessing*, maka ruang pencarian solusi dapat lebih diperkecil.
- Mencari heuristik lain pada permainan *Light Up* yang dapat memberikan hadiah pada agen pencari. Pada skripsi ini, agen pencari hanya diberikan hukuman apabila melanggar aturan permainan atau heuristik tertentu.
- Merancang model yang dapat mencabut peletakkan lampu pada petak. Pada skripsi ini, agen pencari solusi hanya dapat meletakkan lampu pada papan, sehingga apabila peletakkan lampu yang dilakukan oleh agen pencari tidak dapat menghasilkan solusi, maka agen pencari tidak dapat mencabut peletakkan lampu tersebut.
- Merancang model yang dapat mengatur bobot hukuman untuk melakukan pencarian solusi pada setiap jenis papan. Pada skripsi ini, bobot hukuman yang digunakan diatur dengan nilai tertentu. Penentuan bobot hukuman yang tepat untuk setiap jenis hukuman tidak dimungkinkan apabila dilakukan secara manual, hal ini disebabkan karena penentuan konfigurasi bobot hukuman yang tepat memiliki jumlah kemungkinan yang sangat banyak.
- Merancang model yang dapat melakukan pencarian secara cepat. Pada skripsi ini, model membutuhkan waktu yang cukup lama dalam melakukan proses pencarian solusi pada sebuah papan dengan jenis 25×25 *normal*, di mana dibutuhkan waktu sekitar 1–2 jam.

DAFTAR REFERENSI

[1] Smith, J. dan Johnson, A. (2020) Solving the light up puzzle using ant colony optimization. *Journal of Artificial Intelligence*, **10**, 123–137.

[2] Salcedo-Sanz, S., Carro-Calvo, L., Ortiz-García, E. G., Pérez-Bellido, A. M., dan Portilla-Figueras, J. A. (2014) A nested two-steps evolutionary algorithm for the light-up puzzle. *Journal of Heuristics*, **20**, 265–290.

[3] Wang, L., Zhang, S., dan Liu, G. (2015) Solving the light up puzzle using particle swarm optimization. *Expert Systems with Applications*, **42**, 6343–6353.

[4] Mirjalili, S. (2014) Grey wolf optimizer. *Advances in Engineering Software*, **69**, 46–61.

[5] Georgiev, A. (2021) Light up - online puzzle game. https://www.puzzle-light-up.com/. 15 Desember 2022.

[6] Gridin, I. (2021) *Learning Genetic Algorithms with Python: Empower the performance of Machine Learning and AI models with the capabilities of a powerful search algorithm (English Edition)*. BPB Publications, India.

[7] Foundation, P. S. (2021) pickle - python object serialization. https://docs.python.org/3/library/pickle.html.