Dokumentation:

Idee:

Vorher wurde den Workern ein bestimmtes CityTile zugeteilt. Nun soll dem Worker aber eine City zugeteilt werden.

Umsetzung:

Da Citys aber mehrere City\_Tiles haben können, wurde eine Neu Methode (get\_closest\_citytile\_from\_city() ) hinzugefügt, die das am naheliegenste City\_tile zur Unit zurück gibt, zu der die Uni dann gehen soll. So soll ein unnötiger Weg zu einer weiter entfernten City Tile verhindert werden.

Auswertung:

Scheint gut zu funktionieren

Idee:

Der Bauer der Stadt bekommt die Stadt sofort zugewiesen, da ansonsten die neue Stadt nicht mit Rescourcen versorgt wird.

Umsetzung:

Auswertung:

Sub optimal, da dann die alte Stadt keinen Worker mehr hat, da wenn ein neuer Worker gespawnt wird dieser der aktuellen Stadt zugewiesen wird. 🡪 Besser die build\_location verbessern!

Idee:

Nachdem ich festgestellt habe, dass die unit\_to\_city dicts nicht richtig funktioniert haben, und so eine unit immer zur nächstgelegenen Stadt gelaufen ist, habe ich diesen Fehler behoben. Allerdings werden nun kaum noch Städte gebaut. Dies will ich mit geschicktem auswählen eines Bauorts an einem Angrenzenden CityTile machen. Hierbei soll zuerst geprüft werden, ob der Worker schon auf einem Angrenzenden Feld steht, wenn ja, soll dort eine Neue Stadt gebaut werden.

Umsetzung:

Teste zuerst, ob Unit auf Leerem Feld steht, welchen an ein City Tile angrenzt. Hierfür holt man sich die Zelle der Unit und testet, ob diese Zelle Leer ist, wenn Sie leer ist, dann testet man, ob daas Tile an ein CityTile angrenzt, indem man über die CityTiles iteriert und mit der is\_adjacent Methode überprüft, ob sie angrenzend sind. Ist dies der Fall, wird an der Stelle gebaut. Wenn das nicht der Fall ist, soll der Worker in Richtung eines Freien Tiles gehen, welches bebaut werden soll und an die Stadt angrenzt.

Auswertung:

Es wird nun nur noch eine riesige Stadt gebaut. Die verringter den Energieverbrauch allerdings ist es meistens Sinnvoller neue Städte in der näher von aanderen Resourcen zu machen. Mit dieser Version hat sich der Score bei 777 eingependelt, was schon mal sehr gut ist. Jedoch beim Beobachten der Replay habe ich gesehen, dass viele Gegner schneller neue CityTiles bauen und schneller neue worker spawnen und somit schneller arbeiten können. Manchmal wird sogar versucht meine Stadt abzugrenzen, sodass die worker nicht mehr aus einem bestimmten Bereich entkommen können.

BUG:

Der agent ist abgestüzt, weil im endgame keine unzugewiesenen Resourcen mehr vorhanden waren und somit get\_closest\_resource None zurückgegeben hat. Sollte nun keine unzugewiesene Resource mehr vorhanden sein, wir die am nächsten liegende resource verwendet.

IDEE: Die Worker sollen am Anfang schneller bauen, damit wir schneller mehr worker haben. Beim Bauen werden die Worker manchaml blockiert, da sie nicht durch die Städte gehen können, weil sie sonst die Resourcen verlieren, aber blockiert werden von workern, die noch sammeln. Die Unit blockieren sich manchmal an den Resourcen, weil eine Unit auf eine Resource will, wo schon ein worker drauf ist. Dies kann verhindert werden, indem die get closest\_Resource Methode nur Resourcen ausgibt, wo gerade kein worker drauf ist. Allerdings ist hierbei zu beachten, dass im late Game nur noch wenige Resourcen verfügbar sind, also sollten ferne Ressourcen ausgegeben werden, wenn jemand drauf steht aber ganz nahe nicht.

IDEE: Städte sollen geschickt an neuen RescourceFelder angebaut werden.

IDEE:

Beim Replay ansehen war die Startaufstellung so ungünstig, daass das City\_Tile von Resource Tiles umgeben war und somit kein angrenzender Anbau möglich war. 🡪 Methode muss überarbeitet werden möglicherweise auch, wenn die Satdt in der Nacht verschlungen wird aalso der spieler keine Städte mehr hat.

IDEEN VORRAT:

Idee:

Vor erster Nacht wird immer eine Stadt gebaut, welche aber sofort wieder durch zu wenig fuel zerstört wird. 🡪 Keine Stadt bauen wenn den Night Cycle nur 5 Runden entfernt ist?

Gegenargument: Funktionierte bisher aauch so gut und wenn der Baau oft klug gewählt wurde, kann auch der Bau in der Nacht erfolgen. ABER!!! Es könnten auch ganze städte deswegen zerstört werden, wenn durch das neue Gebäude nicht genug Fuel da ist.

Deep Q-Network

<https://www.kaggle.com/polhansen/lux-ai-tensorflow-dqn/edit/run/84349942>

## Analyse des Deep Q-Netzwerks:

### Backbone

We get the map status (of shape (8, 32, 32)) and fit into out backbone model to extract a 64-d state vector.

### Submodels

There is 2 submodels (city tile and worker model, I omit the cart model to ease the training) attach to this backbone which will receive the state vector, game stats and the corresponding state of the unit to determine the action.

### Reward function

The reward function is defined as Reward, 𝑅=Agent's score − (difficulty constant) ∗ opponent's score, where difficulty constant is currently set to 0.1 and Score, 𝑆=3(number of city tiles) + number of units + research points/100 + (turns survived)/360 − (opponent's score)

I think the reward function can be modified or can replaced by a neural network trained based on others' gameplay.

### Replay buffer

We save the tuple (state, action, reward, next\_state) in a buffer for the model to relearn whether an action is good or bad.

Questions asked:

Im bestehenden DQN wird als vor Verarbeitung die state\_map (8, 32, 32) auf eine Shape von (32,32) zusammengefügt. Somit werden alle Dimensionen der State\_map pixelweise zusammen gefügt. Die 8 Dimensionen der state\_map beinhalten: ()

Idee:

Mapabhängige Vorgehensweise:

Bspw wenn sehr kleine Map dann ressourcen sammeln und zum schluss vor der letzt Nacht eine riesige stadt bauen. Und wenn Große Map ist, dann schnellstmöglich Research betreiben.

Einheiten immer in den zuletzt gebauten CityTile spawnen, da dort noch immer ein Gebiet von relevanz ist, was bei den alten CityTiles vlt nicht mehr der Fall ist.