[Note du traducteur : ceci est une traduction en français libre et non officielle du FFF paru sur le forum]

Friday Facts N°331
Version 0.18.0 et modifications de la recherche de trajectoires des trains

Posté par Klonan, V453000 et boskid

le 24/01/2020

Sortie de la version 0.18.0 (par Klonan)

En début de semaine, nous avons appuyé sur le bouton de déploiement de la version 0.18.0 (voir les modifications). Cela a été une surprise pour beaucoup de nos joueurs, car en général, le délai entre les versions majeures est plus long et le contenu de ces mises à jour est plus important. Cependant, ce n'est pas comme au bon vieux temps, nous essayons de garder la taille des publications aussi petite que possible (FFF-314).

Cela signifie que ce qui est actuellement dans la version 0.18 n'est qu'une toute petite partie du travail à faire sur la version 0.18, et les versions des mois à venir continueront à compléter notre liste de tâches sur la version 0.18.

Lorsque tout ce qui figure sur la liste pour la 0.18 sera terminé et que le moment sera venu, nous transformerons la version 0.18 en version 1.0.

Ce que nous avons accompli avec la version 0.18.0 :

- Interface graphique
  - Refonte du menu principal
- Graphisme
  - Animation de l'eau
  - Animation des arbres
  - Correction des couleurs (table de correspondance)
  - Nouvelles explosions et effets des dommages
- Autres
  - Optimisations
  - Nouveau système pour les particules
  - Premiers travaux sur une nouvelle conception sonore
  - Connexion à Steam

Ce qu'il nous reste à faire dans la version 0.18 :

- Interface graphique
    - Interface graphique des personnages
    - Bibliothèque de blueprints
    - Interface graphique des statistiques (production, statistiques du réseau électrique, etc.)
    - Interfaces graphiques des entités (bras, machines d'assemblage, coffres, etc.)
    - Interface graphique de l'écran principal (console, minimap, etc.)
    - Et bien d'autres encore...
- Graphisme
    - Refonte des pompes côtières
    - Refonte des machines d'assemblage
    - Refonte des diffuseurs de modules
    - Icônes en haute résolution
    - Dernières retouches et polissage
- Autres
    - Nouvelles améliorations de la conception sonore
    - Mini-tutoriels
    - Remplacer la NPE [NdT : Expérience du Nouveau Joueur] par l'ancien tutoriel
    - Equilibrage final du jeu et ajustements
    - Finalisation des traductions et relecture

Dans cette optique, il ne serait pas logique de marquer la 0.18 comme stable avant que la plupart des éléments ci-dessus ne soient terminés. Nous avons fait de la version 0.18 une version majeure parce qu'elle va casser des mods avec tous les changements que nous faisons, et bien qu'au départ elle n'en ait pas tant cassé que ça, beaucoup de choses à venir auront un plus grand impact, comme l'interface graphique des personnages.

Interface graphique des personnages ?

Au départ, nous avions prévu que l'interface graphique des personnages serait dans la première version de 0.18. Cependant, la tâche s'est avérée assez difficile dans la manière dont elle a été écrite, et lorsque la date de sortie est arrivée, elle n'était pas prête, nous avons donc retardé la parution de la 0.18.0 (deux fois). Après un examen difficile et approfondi, nous avons décidé de nous débarrasser de la plupart du travail et de prendre un nouveau départ avec un autre membre de l'équipe. Avec la programmation, il faut savoir quand tenir et quand plier.

Avec ce changement, nous avons décidé de sortir la version 0.18 sans l'interface graphique des personnages. L'alternative était un délai supplémentaire de 4 à 6 semaines, ce qui, avec seulement 8 mois restants, est une grosse perte de temps. Nous ne voulons pas non plus reprendre la vieille habitude de toujours retarder la sortie pour une raison ou pour une autre, et les choses que nous présentons dans les Friday Facts ne pouvant être vues en jeu avant des mois et des mois.

Après cette décision, Dominik, qui travaillait sur l'interface graphique des personnages, a décidé de quitter l'équipe de Factorio.

La Campagne est abandonnée (par V453000)

Dans le récent FFF-329, nous avons mentionné deux approches différentes de la Campagne. Dans le présent FFF, nous annonçons que la Campagne a été supprimée, et je vais essayer d'expliquer pourquoi.

Après la suppression du tutoriel/NPE, j'ai réalisé plusieurs choses et cela m'a fait réévaluer ce que la Campagne essayait d'être, en posant des questions comme :

- La Campagne est-elle censée être "La Voie", comment jouer le jeu ou simplement du "contenu supplémentaire" ?
- La Campagne devrait-elle essayer d'imiter le jeu libre ?
- Le fait de perdre de la progression et de repartir de zéro est-il un problème à ce point important ?
- …

À mon avis, si nous essayons de créer "La Voie", la façon dont Factorio est censé être joué, il serait logique d'essayer de rester très proche de la façon dont le jeu libre fonctionne, puisque c'est "La Voie" depuis de nombreuses années maintenant. Cependant, essayer d'imiter le jeu libre signifie intrinsèquement qu'il ne sera jamais identique au jeu libre, tout en ajoutant certainement de nouveaux problèmes (et, espérons-le, des avantages).

Essayer d'être similaire au jeu libre et créer une représentation de la façon dont Factorio "devrait être joué" était l'état d'esprit derrière la Campagne en expansion continue. Le principal avantage par rapport au jeu libre est que le contenu est segmenté en petits morceaux (ce qui est déjà le cas en raison de la production des packs scientifiques et de la progression technologique - le joueur ne peut pas tout construire dès le départ). Cependant, avoir une quête à court terme, se trouver dans une zone limitée de la carte, interagir avec des ruines d'usines existantes qui empêchent le joueur d'aller dans la zone suivante, etc. entraîne beaucoup de problèmes inattendus. En comparaison, ce n'était pas très amusant à jouer.

Avec cela, le jeu libre avec toute sa liberté sera probablement toujours la meilleure façon de vivre Factorio. Il serait très logique (pour moi) de s'efforcer plutôt de créer un ensemble de petits scénarios comme contenu secondaire pour donner une expérience nouvelle après 2 000 heures de jeu libre. Étendre Factorio avec des scénarios séparés, au lieu d'essayer de réinventer Factorio avec cette Campagne qui se déploie.

Étant un contenu secondaire, je pense qu'il serait beaucoup plus acceptable de perdre de la progression entre les niveaux si cela crée un nouveau scénario ou un défi intéressant. La raison pour laquelle nous avons choisi le concept de carte expansive était de ne jamais forcer le joueur à perdre des progressions et à reconstruire sa base. Si cette restriction est levée, nous pourrions répartir les scénarios sur plusieurs niveaux. Pendant les vacances de Noël, j'ai essayé de recycler la carte en expansion actuelle en un ensemble de niveaux séparés, et nous avons même mis en place une version jouable mais très rudimentaire de ces niveaux pendant la première semaine de janvier.

Nous nous sommes réunis la semaine dernière pour reconsidérer ce qu'il fallait faire de la Campagne parmi les deux options du FFF-329. La Campagne en expansion nous semblait trop risquée en raison de tous ses problèmes, et nous n'en étions pas assez sûrs. Les niveaux séparés seraient une option plus sûre, mais s'ils ne sont pas la principale façon de jouer au jeu, alors ils ne sont pas aussi importants que le cœur du jeu pour la version 1.0.

Il est encore possible que nous ajoutions une Campagne après le lancement de la version 1.0, mais pour l'instant, nous pensons que l'abandon de l'idée d'une Campagne nous permet de nous concentrer sur le cœur du jeu pour les mois à venir.

Changements dans la recherche de trajectoires des trains (par boskid)

Aujourd'hui, je voudrais parler des changements apportés à l'outil de recherche de trajectoires des trains que j'ai réalisé pour la 0.18. Il s'agit de changements subtils visant à corriger des cas particuliers bizarres, mais d'abord un petit glossaire :

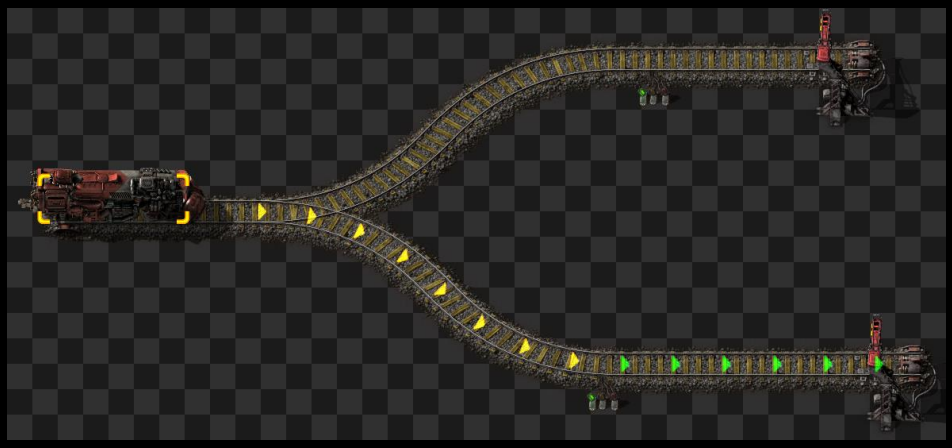
- Rail : une entité unique sur laquelle les trains peuvent circuler. L'unité de base de la construction.
- Segment : une série de rails qui ne sont pas séparés entre eux par des jonctions, des signaux ou des gares. L'unité de base pour la recherche de trajectoires.
- Bloc : un groupe de segments qui se rencontrent ou sont reliés entre eux sans signaux. L'unité de base pour la réservation des trains.

En 0.17 et antérieurement, la recherche de trajectoires utilisait la méthode classique de Dijkstra pour trouver le chemin avec la plus petite pénalité. Les nœuds sont associés à des segments avec des informations supplémentaires sur la direction du déplacement. Les bords sont associés à des connexions de segments (transitions).

Problème 1 : la longueur du dernier segment n'est pas comptée dans la pénalité

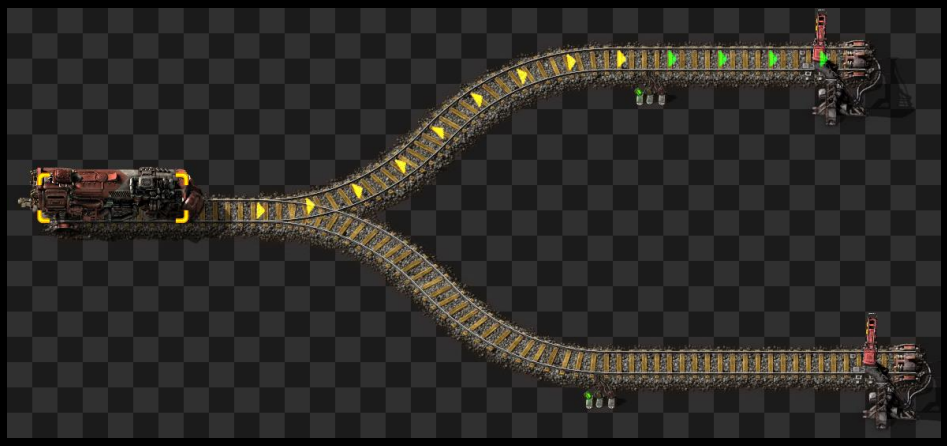
C'est ce problème qui m'a poussé à me pencher sur le code la recherche de trajectoires. Les arrêts de train sont toujours à la fin d'un segment. Les segments peuvent avoir au maximum 2 arrêts de train, un pour chaque sens de déplacement. Il est impossible d'en ajouter d'autres, car l'arrêt de train à l'intérieur diviserait le segment en deux. Cela signifie qu'un train qui entre dans le dernier segment de son trajet doit parcourir toute la longueur pour atteindre sa destination.

Lors de sa mise en œuvre, le coût total sur les nœuds a été calculé comme le coût de tous les segments et transitions précédents. Cela signifie que le nœud avec l'arrêt de train demandé serait choisi dans une file de priorité avant que le coût de la longueur du dernier segment ne soit ajouté. Comme le nœud a l'arrêt de train demandé à son extrémité, le trajet serait attribué même s'il y a d'autres nœuds dont le coût actuel est plus élevé, mais qui serait quand même meilleur puisque le coût du dernier segment serait inférieur. Le coût du segment actuel serait ajouté aux nouveaux nœuds lors de l'expansion, mais il était déjà trop tard - le trajet a déjà été fourni.



0.17 – La recherche de trajectoire choisit le chemin du bas qui est le plus long

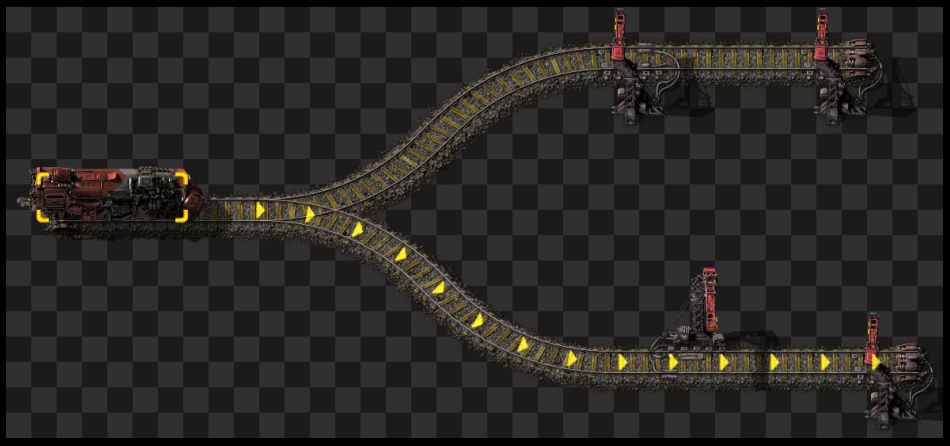
La solution était assez simple : inclure la longueur du segment du nœud dans le coût total du nœud depuis le début. Cela a changé le code d'expansion du nœud car il n'ajoutait pas le coût du segment du nœud actuel, mais celui du segment du nœud suivant. Cela conduirait à des cas où la longueur du premier segment ne serait pas comptée, j'ai donc ajouté un code qui, lors de la création des nœuds de début, les crée avec la pénalité du segment de début entier.



0.18 – La recherche de trajectoire corrigée choisit le chemin du haut qui est le plus court

Problème 2 : la station opposée dans le dernier segment n'est pas comptée dans la pénalité

La deuxième question est également liée au dernier segment et a une explication similaire à celle du point ci-dessus. Le coût de l'arrêt de train dans le segment actuel n'a été ajouté que lors de l'extension du nœud aux segments suivants.

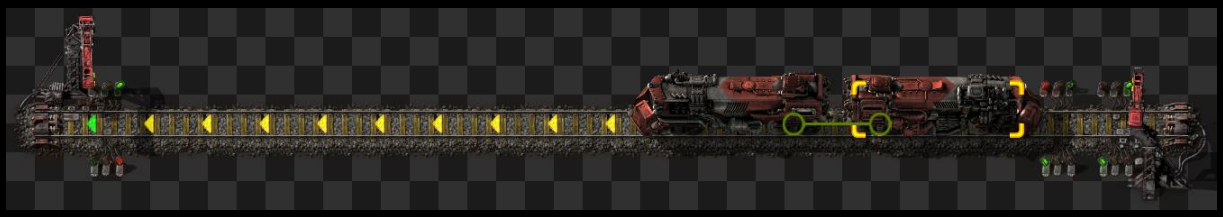


0.17 - La recherche de trajectoire choisit le chemin du bas car il ne contient pas la pénalité de la station opposée

La solution simple consistant à déplacer la pénalité d'arrêt de train d'un segment (en cas d'extension, ajouter le coût des arrêts de train au segment suivant, et non au segment actuel) et à inclure les deux arrêts de train à l'intérieur du segment de début a réglé le problème (ce n'est pas la solution finale comme ce sera expliqué dans le point 5).

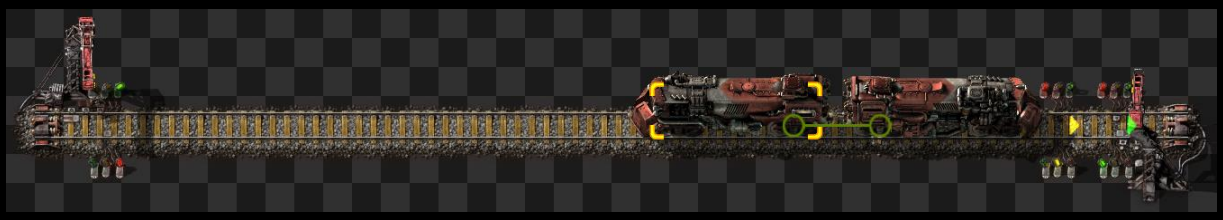
0.18 - La recherche de trajectoire corrigée choisit le chemin supérieur le plus court. Les deux chemins ont une pénalité de 1 station.

Problème 3 : la longueur du premier segment ne prendrait pas en compte la position du train
Maintenant, après la correction du problème 1, j'ai remarqué que l'ajout de la pénalité de tout le segment de début a une faille - si un train peut aller dans les deux sens et que les deux extrémités du train sont dans le même segment, le coût du premier segment serait le même dans les deux sens et s'annulerait.



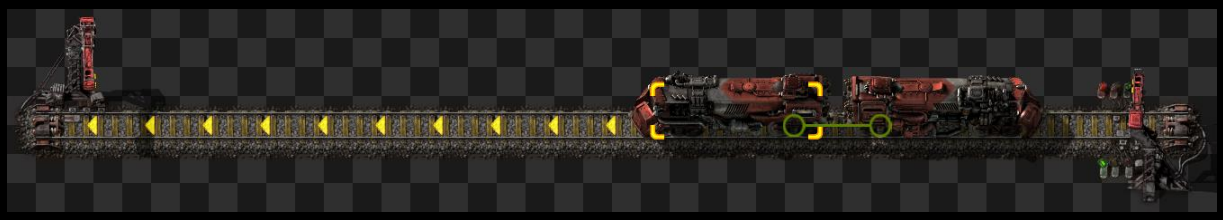
0.17 - La recherche de trajectoire choisit d'aller à gauche. Le chemin vers la droite a un coût supérieur de 2.

La solution était simple : puisque la recherche ne connaît que le rail de début dans le premier segment, il doit passer par tous les rails dans une direction donnée pour trouver le rail de fin dans un segment donné. J'ai ajouté la mesure de la longueur pendant cette recherche et l'ai utilisée comme coût initial. Ainsi, la position du train aura une incidence sur le coût initial lorsque l'on regarde les trajets dans les deux directions.



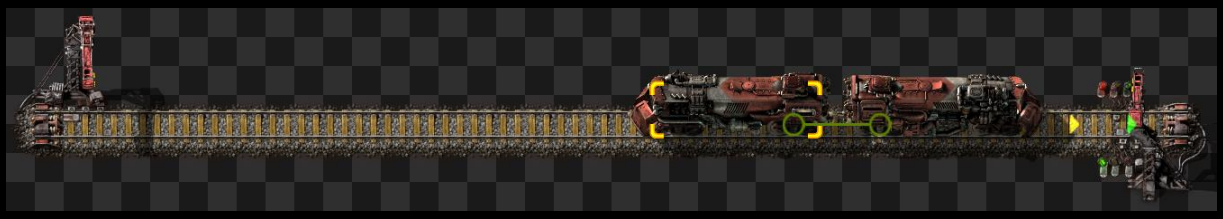
0.18 - La recherche de trajectoire corrigée choisit de se diriger vers la droite. La position du train est prise en compte.

Problème 4 : un trajet à segment unique a la priorité sur les trajets à segments multiples
Maintenant que le problème 3 a été résolu, il se trouve qu'il y avait un code supplémentaire pour gérer les trajets d'un seul segment. Il couvre des cas bizarres tels que le segment cyclique, ou quand la gare est dans le même segment. Ce code calcule le coût du chemin en fonction des rails (et non des segments) et s'il trouve un chemin, il choisira le trajet le moins cher et sautera la logique principale de recherche de trajectoires pour les chemins multi-segments.



0.17 - La recherche de trajectoire choisit le chemin d'un seul segment.

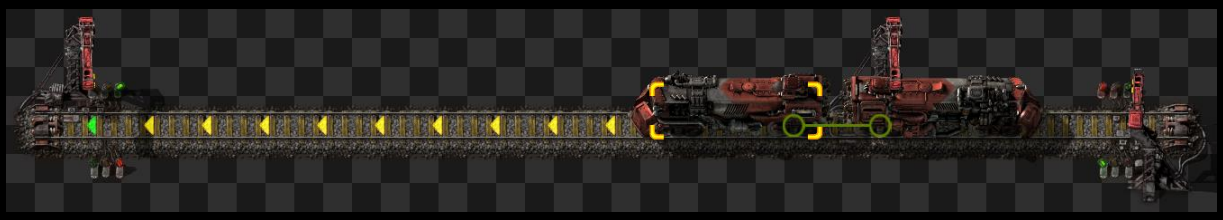
La solution : s'il existe un chemin à segment unique, créer un nœud marqueur avec le coût du chemin à segment unique le plus court et le jeter dans la file prioritaire de la recherche de trajectoire. Si ce chemin à un seul segment est vraiment le plus court, alors ce nœud marqueur sera sélectionné lors de l'expansion, ce qui signifie que tous les chemins à segments multiples ont un coût plus élevé. S'il y avait un chemin multi-segments plus court, ce nœud marqueur ne serait pas choisi et serait donc ignoré.



0.18 - La recherche de trajectoire corrigée choisit la voie multi-segments parce qu'elle est moins coûteuse.

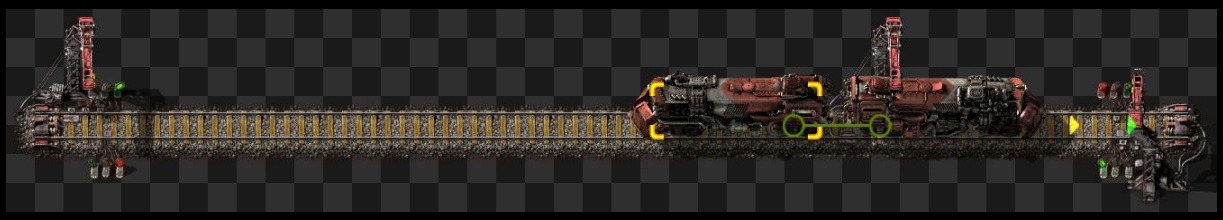
Problème 5 : la station opposée dans le premier segment a été ajoutée à la pénalité

Après la correction du problème 2, chaque segment du trajet comportant des arrêts de train ajoutait la pénalité des arrêts de train, sans prendre en compte s'il s'agissait d'une entrée ou d'une sortie du segment. Cela signifie qu'un train à double sens dont une extrémité se trouve à l'intérieur du segment avec un arrêt de train qui se trouve sous le train, considérerait toujours cet arrêt de train dans la pénalité.



0.17 - La recherche de trajectoire choisit le chemin de gauche puisque le chemin de droite a la pénalité de l'arrêt de train

Sur cette base, j'ai décidé de supprimer la pénalité des arrêts de train en sens inverse dans le premier segment. Les mesures de performance effectuées ici ont révélé un autre problème : la correction du problème 2 était défectueuse. Il ne fallait pas ajouter la pénalité de l'arrêt de train dans le dernier segment, car cela obligeait la recherche de trajectoire à agrandir les nœuds jusqu'au coût de l'arrêt de train en cherchant d'autres trajectoires qui n'auraient peut-être pas cette pénalité de dernière gare (alors qu'en fait, c'est inévitable). J'ai donc vu que l'arrêt de train opposé dans le premier segment et l'arrêt de train de fin dans le dernier segment ne doivent pas ajouter de pénalité. Cela a donné la solution finale pour la pénalité d'arrêt de train : lors de l'expansion d'un nœud, l'arrêt de train de sortie dans le segment actuel ajoute une pénalité et l'arrêt de train opposé dans l'entrée du segment suivant ajoute une pénalité.



0.18 - La recherche de trajectoire corrigée choisit le bon chemin. L'arrêt sous le train n'est pas compté.

Problème 6 : la longueur du bloc à partir du début n'a pas été correctement mise à jour

C'est une question assez subtile. Elle est liée à cette règle de pénalité :

*"Lorsque le bloc est occupé par un train => Ajouter une pénalité de 2 * longueur du bloc divisée par la longueur du bloc depuis le début, de sorte que les trajets lointains occupés n'ont pas beaucoup d'importance."*

Cela signifie que pour la recherche de trajectoires, non seulement le coût du début est compté, mais aussi le nombre de blocs depuis le début - chaque fois qu'un nœud s'étend à un autre segment qui provient d'un bloc différent du segment du nœud précédent, augmente le nombre total de blocs depuis le début de 1. Cela fonctionnait bien tant que l'expansion du nœud créait un nouveau nœud. Dans le cas de la mise à jour d'un nœud, le nombre total de blocs n'était pas mis à jour par rapport à l'ancienne valeur laissée dans le nœud. Cela signifie que le nœud mis à jour aurait le coût d'un nouveau chemin moins coûteux, mais aurait la longueur de bloc du début du chemin précédent.

0.17 - La recherche de trajectoire choisit le chemin du bas parce que le `blocDistanceFromStart` n'est pas correctement mis à jour dans la partie supérieure.

Dans ce dispositif, le problème se pose dans la partie supérieure. Le segment sous la locomotive du milieu en haut est d'abord élargi en partant du segment droit sur sa gauche - c'était moins cher. Maintenant, ce segment a le coût de la ligne droite et la longueur du bloc à partir du début de 1, puisqu'il n'y avait qu'une seule transition vers un autre bloc. Il y avait aussi la pénalité de l'occupation du bloc par un autre train.

Lorsque l'extension passe par le rail supérieur, elle touche 4 signaux ferroviaires et la longueur du bloc à partir du début est donc augmentée à 4. Ce chemin a un coût plus élevé jusqu'au signal ferroviaire de gauche près de la locomotive centrale du haut. Ici, cependant, le coût du bloc occupé par un autre train est plus faible puisque nous entrons dans le 5ème bloc dès le début en passant par le chemin supérieur. Le chemin latéral est choisi pour mettre à jour le coût du nœud lié au segment avec la locomotive du milieu en haut, mais la distance du bloc depuis le début n'a pas été mise à jour. De ce fait, la pénalité sur le prochain signal ferroviaire qui va vers un autre bloc occupé, était égale à la distance du dernier segment divisée par 2 au lieu de 6. Cette différence rendrait le chemin à travers la partie inférieure moins coûteux puisque le chemin droit est ici coupé et que le nœud pour le segment sous la locomotive du milieu est créé avec la bonne distance de bloc à partir du début.

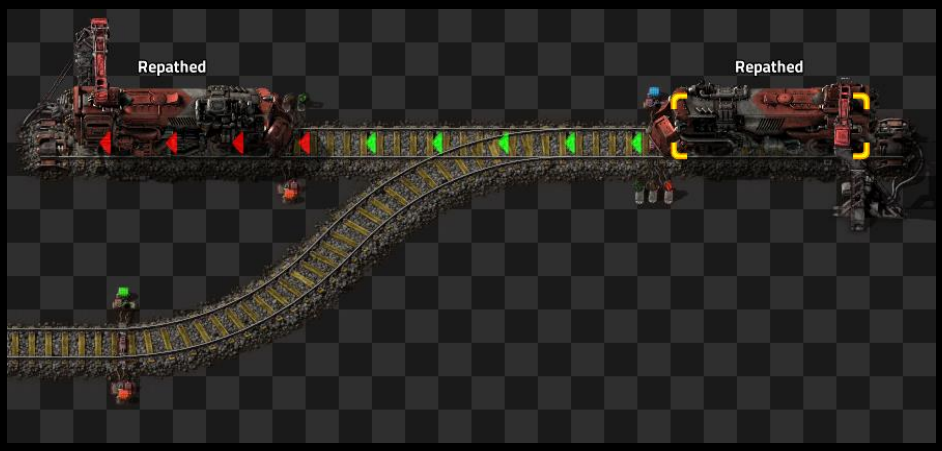
La solution était simple : en modifiant le coût de début sur un nœud existant, il fallait également modifier la longueur du bloc de début.

0.18 - La recherche de trajectoire corrigée choisit la voie supérieure

Problème 7 : la nouvelle recherche de trajectoires permettrait de vider le compteur de ticks d'attente du train à un signal.

Ce n'est pas exactement un problème de recherche de trajectoires, mais cela conduit à des blocages de trains. Si un train attend un signal, il répète périodiquement sa recherche pour trouver un autre chemin possible qui peut maintenant être débloqué. La nouvelle recherche efface cependant le compteur de la durée d'attente du train au signal. La logique des nouvelles recherches périodiques en était consciente, de sorte qu'elle restaurait le compteur d'attente au signal précédent lorsque le train entrait en état d'arrivée au signal. Cependant, un correctif a modifié le comportement du train, et le train ne passe plus par l'état d'arrivée au signal, ce qui a permis d'effacer le compteur de ticks. Cela a enfreint la règle suivante :

"Lorsque le trajet comprend un train en attente à un signal ferroviaire => Ajouter une pénalité de 100 + 0,1 pour chaque tick que le train a déjà attendu".



0. 17 - Le train de droite est bloqué parce que la pénalité du train de gauche, qui dépend du temps, est réinitialisée à chaque fois que le train de gauche répète sa recherche de trajectoires.

Solution universelle : ne vider ce compteur que si le train change d'état pour autre chose que d'attendre à un signal.

0. 18 - Les ticks d'attente corrigés sur la logique des signaux, après 3 nouvelles recherches la pénalité du train de gauche augmente jusqu'à un point où le train de droite choisit le long chemin.

Optimisation de la recherche de trajectoires - file prioritaire

Au cœur de la recherche de trajectoires se trouve une file prioritaire qui rassemble les nœuds ouverts. Comme elle a été mise en place jusqu'à présent, la file prioritaire était basée sur une liste à double lien. La recherche du nœud le moins cher (priorité la plus élevée) était rapide (constante), mais l'insertion de nouveaux nœuds ou la mise à jour de nœuds existants était, dans le pire des cas, linéaire (O(n)). Après une phase de prospection rapide, j'ai décidé d'implémenter un empilement binaire avec des propriétés minimales sur un tableau. Ce changement à lui seul a permis d'accélérer de 20 % la recherche.

Optimisation de la recherche de trajectoires - heuristique (conversion de Dijkstra en A*)

La deuxième optimisation appliquée a été l'ajout d'une fonction heuristique qui donne le coût minimum pour chaque fin. Cela signifie que l'expansion des nœuds est maintenant guidée vers l'objectif par la fonction heuristique, et que moins de nœuds devraient donc être visités avant de trouver l'objectif. Cela a permis une nouvelle augmentation des performances.

L'impact des changements de la recherche de trajectoires des trains, c'est que cela devrait être environ deux fois plus rapide et comporter moins de cas bizarres.

Comme toujours, faites-nous savoir ce que vous en pensez sur notre forum.

Discuter sur nos forums

Discuter sur Reddit

[NdT : Traduit avec l'aide de www.DeepL.com/Translator (version gratuite)]