Dungeon2 解説

問題概要

- 無向グラフがあります
- グラフを探索して，構造を決定してください
  - 各頂点からは，各辺に番号が付けられている
- 探索するために使えるものは，
  - 現在地から出ている辺の本数がわかる
  - 辺をたどって移動することができる
  - 最後にたどった辺が，現在地から何番目の辺かわかる
  - X 色の宝石を使って，頂点に情報を持たせておくことができる

構造決定？

- 別に必ずしもグラフの構造を決定する必要はない
- が，求める値は「最短距離が D の頂点対が何個あるか？」

- 構造を決定せずにこの値を求めるのは大変そう
- なので，グラフの構造を決定することを考える

「最短距離が D の頂点対が何個あるか？」

- 最短路問題
- 全点間の最短路を求めましょう
- N <= 200 と小さいので大抵の方法が使えます

- Warshall-Floyd が実装簡潔，定数倍高速でおすすめです
  - O(N^3)，余裕

使える情報

- X 色の宝石
  - かなり重要そう
- 頂点から出ている辺の数（次数）
  - 使える？？
  - どうせ「全頂点の次数が同じ」みたいな悪質なケースが入ってるんだろうなあ

深さ優先探索

- 頂点に値を持たせられる(しかも初期値が決まっている)ので,「その頂点をすでに訪問した」というのは判定できるようになる
- 「どの辺からやってきたか」も常にわかる
  - 移動元へ戻ることは可能
- 深さ優先探索 (DFS) ができそう？

小課題 1

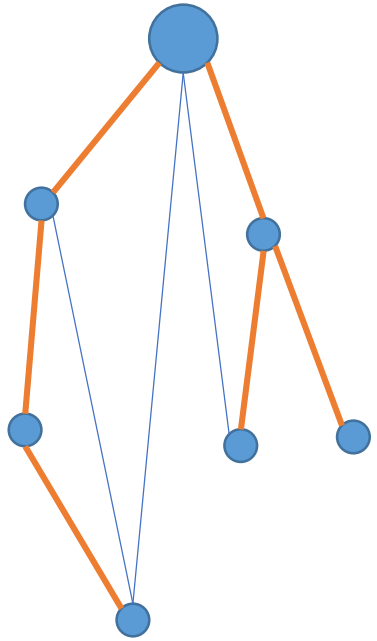
- DFS を行う
- 頂点の値は次のようにする
  - 1: まだ訪れていない (初期値)
  - それ以外: すでに訪れた (異なる頂点には異なる値を割り当てる)
- 各頂点では，戻る方向以外のすべての辺をとりあえずたどってみて，
  - 未訪問だったら再帰的に探索
  - 訪問済みだったら，割り当てられてる値を見て，辺を張る
  - すべての辺をたどったら，戻る方向の辺を使って元の頂点に戻る
- X=100 なので余裕
- 小課題 1 が解けて，17 点が得られる

DFS 木

- この探索を行うときに，辺は 2 種類に分かれる
  1. 探索中の移動時に，移動先が未訪問となる辺
  2. 探索中の移動時に，移動先が訪問済みとなる辺（後退辺）
- 1. の辺だけを見ると全域木になっている

DFS 木の性質

- DFS の開始点を根として根付き木にして考える
- すると，2. の辺は，ある頂点からその先祖の頂点を結ぶ辺になることがわかる



DFS 木の構成

- 小課題 2 以降では X=3 なので，3 種類の状態しか頂点に持たせられない
- が，訪問済みかを覚えるだけなら十分

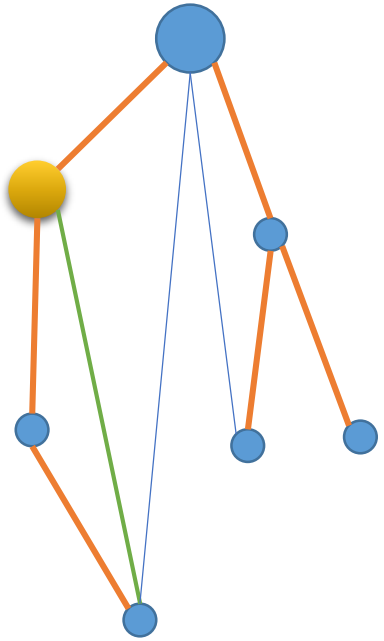
- 小課題 1 と同様の方法で行える
  - ただし，辺をたどった先がすでに訪問済みだった場合でも，具体的にどの頂点につながっているのかは分からない
  - 後退辺が何かある，ことだけ覚えておく

小課題 2

- 後退辺の行先が決定できればよい
- 後退辺があったら，後退辺の行先の頂点にマークをつける
- 全域木を根まで遡って，マークが出てくるかどうか探す
  - マークがあれば，そこが後退辺の行先
  - マークがなければ，後退辺は葉方向へ延びているので，後退辺を逆向きに見るのに任せる
- 遡った後，マークを消すため再び後退辺をたどる
- 状態は 3 つあれば足りる
  1. 未訪問，マークなし
  2. 訪問済，マークなし
  3. 訪問済，マークあり（未訪問，マークなしは DFS をしている限り現れない）

マーク

- 緑の辺を下の頂点から見る場合は，上の頂点にマークをつけて，木の上をたどってマークを探索



小課題 2

- 移動回数を見積もる
  - DFS 木の移動のために 2(N - 1) 回
  - 後退辺は M – N + 1 本ある
  - 各後退辺ごとに,
    - マークして戻るのに 2 回
    - マークを探すのに最大 2(N – 1) 回
    - マークを解除するのに 2 回
    - よって最大 2N + 2 回
  - よって 2(N – 1) + (M – N + 1)(2N + 2) 回以内で終了
- 小課題 2 は解けて合計 44 点が得られる

後退辺の行先の決定

- 各辺について，最悪根まで戻るというのはあまりに効率が悪い
- もっと効率よく決定したい

- ところで，後退辺の行先は，行先の「根からの深さ」さえ分かれば十分
  - 自分より深いところにあることが判明したら，逆向きで見るのに任せる
- 小課題 2 の解法の「マーク」がなければ，使う状態は 2 通りのみで余っている
  - 「訪問済み」のために 2 種類の状態を覚えられる

深さ情報の分散

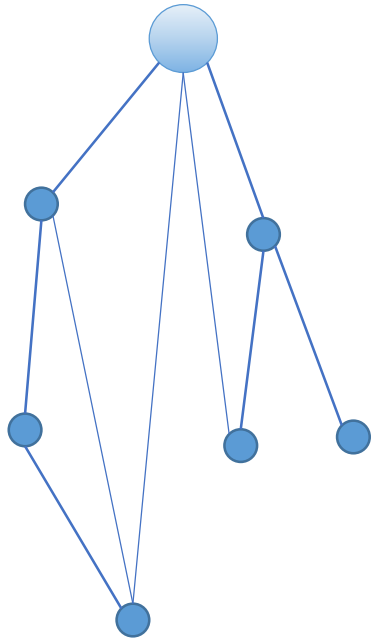
- 深さ情報は 8bit で表現できる
- 「訪問済み」のために使える 2 状態で，深さ情報を少しずつ伝える
  - DFS を 8 回行う
  - i 回目には，深さ情報の i ビット目を覚えておく

小課題 3 (L = 64)

- DFS を 8 回行い，i 回目には根からの深さの i ビット目を計算する
- 具体的には，
  - DFS で新たな頂点を訪問したときには，深さに応じてその頂点に与える値を計算
  - 辺の行先がすでに訪問済みだったら，行先の値を見て，行先の深さ情報を追加
  - DFS の帰りがけにも頂点の値は変更しない
- DFS を行うと頂点の値がめちゃくちゃになる (すべて初期値 1 以外になる)
  - DFS 後に，値消去用の DFS を走らせて，すべて初期値 1 にする
- 複数回 DFS を行うけど，構造の一貫性は大丈夫？
  - 同じ方法で DFS を行えば大丈夫

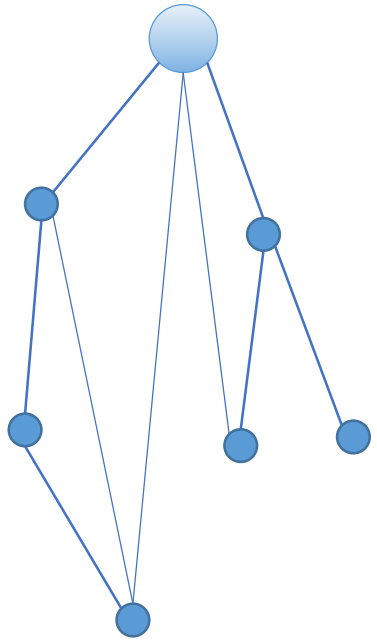
DFS の様子

- 最下位ビットを決定する DFS
- 現在一番上の頂点にいます



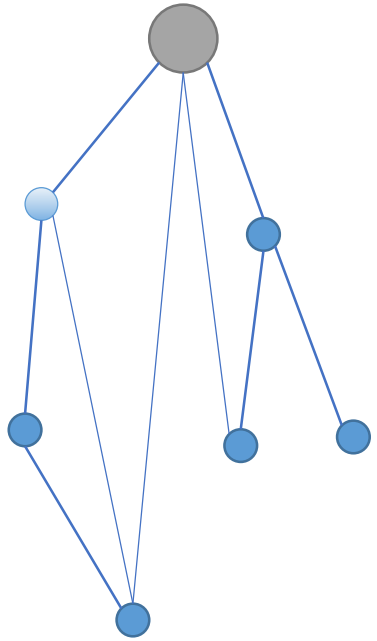
DFS の様子

- 出ている 3 本の辺のうち 1 本選んで移動
- 移動する前に，頂点を「深さ情報：0」の色に更新



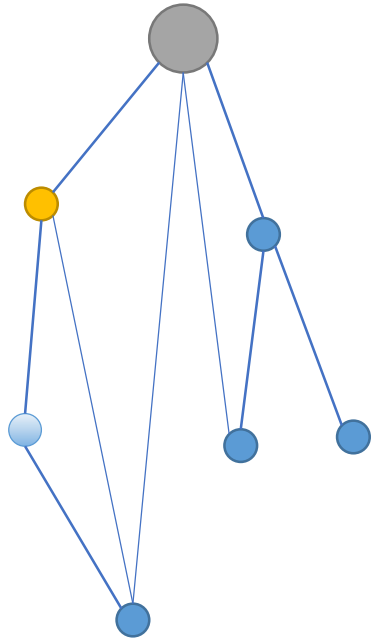
DFS の様子

- 今度は，深さが 1 なので 1 の色にする



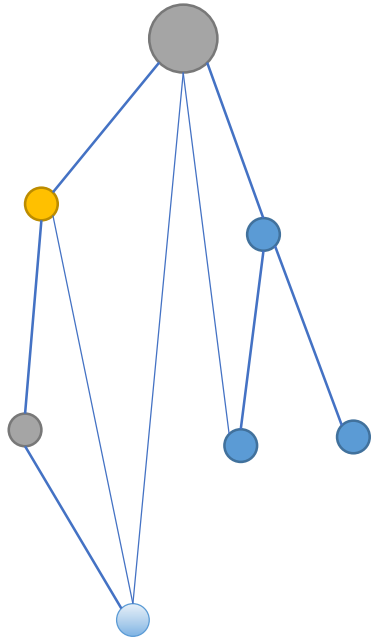
DFS の様子

- 今度は，深さが 2 なので 0 の色にする



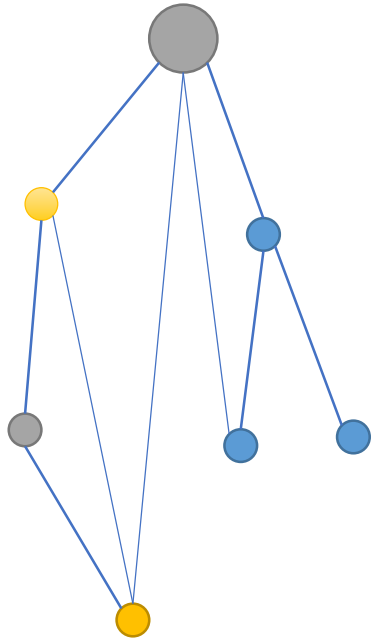
DFS の様子

- 今度は，深さが 3 なので 1 の色にする



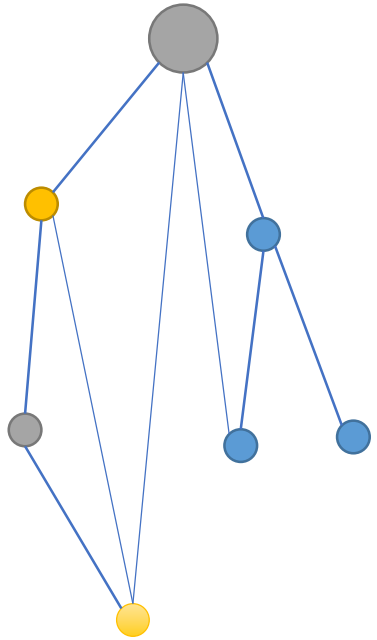
DFS の様子

- 深さが 4 なので，と思いきや
- 色が「深さ情報：1」なので訪問済み
- 同じ色のまま引き返す



DFS の様子

- この頂点からの 2 本目の辺が，「深さの最下位ビットが 1」であると覚えておく



小課題 3 (L = 64)

- 移動回数を見積もる
  - 全域木の辺 N – 1 本は各 DFS で 2 回 (1 往復) 通る
  - 後退辺 M – N + 1 本は各 DFS で 4 回 (2 往復) 通る
  - よって各 DFS で 4M – 2N + 2 回移動
  - 値初期化用の DFS も同じ移動回数
- DFS は 8 + 8 回走るので，合計 64M – 32N + 32 < 64M 回の移動
  - 小課題 3 で L = 64 の場合が解け，66 点が得られる

小課題 3 (L = 60)

- 値消去用の DFS は，最後の DFS の後には不要
- DFS は 8 + 7 回走るので，合計 60M – 30N + 30 < 60M 回の移動
  - 小課題 3 で L = 60 の場合が解け，68 点が得られる

値の消去？

- 値の消去ごときに 28M 回も使うのはバカバカしい
- うまく値を書き換えることで，値の消去を不要にできないか？

小課題 3 (L = 32)

- 帰りがけに，値をすべて (深さの i ビット目: 1) に対応する状態に変更
- すると，DFS 終了時には「根を除いて」すべて同じ値になる
  - 次回の DFS では，初期値を別な値と思って実行
- 後退辺のうち，葉に向かう向きを見ると変なことが起きないか？
  - 深さが見かけ上，必ず 255 (0b11111111) になっている
  - これは無視することにすれば OK
- すると，DFS の回数を 8 回に減らすことができる
- L = 32 の場合が解け，82 点が得られる

全域木

- 1 回 DFS すると，各頂点に勝手な番号が割り当てられる
- 各頂点から出ている各辺が次のいずれかなのかもわかる
  - 全域木の辺，葉へ向かう方向
  - 全域木の辺，根へ向かう方向
  - 後退辺
- 1 回 DFS してしまえば，全域木をたどるのは頂点の値すら見ずに可能

小課題 3 (L = 24)

- 1 回 DFS して，全域木の構造を把握
- 2 回目以降の DFS では，頂点に深さ情報のために 3 通りの値を持たせ，後退辺の情報を取得
- 3 ^ 5 > 200 なので，DFS は 1 + 5 回で十分
- 各 DFS は 4M 回以下の移動で行えるので，全体で 24M 回
- L = 24 の場合が解け，90 点が得られる

後退辺 2 往復？

- 後退辺は，根へ向かう向きのとき以外は情報を得られない
- 無駄な後退辺の移動を減らしたい

- DFS 1 回使うと，各後退辺の方向もわかる

後退辺の向きを知る DFS

- 3 通りの状態
  1. 未訪問
  2. 訪問済みだが，完了していない
  3. 訪問済み，しかも完了済み（帰りがけにこの状態に変更）
- DFS 中では，行先の状態に応じて辺の種類が決まる
  1. 全域木の辺
  2. 後退辺（根へ向かう向き）
  3. 後退辺（葉へ向かう向き）

満点解法

- まず，前スライドの方法で，各辺の種類を把握
  - 以降の DFS では，後退辺も 1 往復ずつしか見ない
- 2 回目以降の DFS で，各頂点に，根からの深さについての 3 通りの状態を持たせる
- DFS は 1 + 5 回
  - 最初の 1 回は 4M 回以下の移動
  - あとの 5 回はそれぞれ 2M 回以下の移動
- 14M 回以下の移動で済み，満点が得られる

得点分布