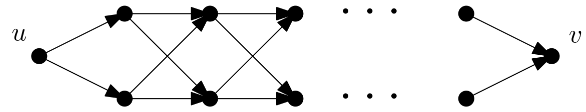
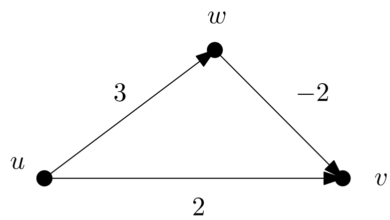
1. *Mnoho cest: Ukažte, jak pro libovolné n sestrojit graf na nejvýše n vrcholech, v němž mezi nějakými dvěma vrcholy existuje $2^{\Omega(n)}$ nejkratších cest.*



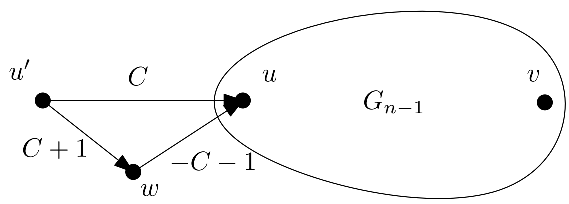
V grafu na obrázku mají všechny uv-cesty stejnou délku a jsou tedy nejkratší. Kolik tedy pro n vrcholů obsahuje nejkratších uv-cest?

2. *Dijkstra I.: Najděte (orientovaný ohodnocený) graf s právě jednou zápornou hranou a bez záporného cyklu, na němž Dijkstrův algoritmus selže (tj. v závislosti na implementaci buď otevře vrchol opakovaně nebo nenajde nejkratší cestu).*



Graf na obrázku je zjevně acyklický (tudíž neobsahuje záporný cyklus). Rozmyslete si, jak bude vypadat běh Dijkstrova algoritmu spuštěného z vrcholu u.

3. *Dijkstra II.: Ukažte příklad grafu s celočíselně ohodnocenými hranami, na kterém Dijkstrův algoritmus poběží exponenciálně dlouho.*



Náznak řešení pomocí konstrukce indukcí: pokud už máme graf G_{n-1} na kterém při hledaní nejkratší uv-cesty stráví Dijkstrův algoritmus exponenciální čas uvažme konstrukci na obrázku. Při spuštění z vrcholu u' Dijkstra nejprve navštíví u a poté pokračuje stejně jako by hledal cestu v G_{n-1}. Jak musíme nastavit konstantu C, abychom vrchol w zpracovali až po dokončení celého výpočtu v G_{n-1}? A proč potom proběhne celý výpočet v G_{n-1} ještě jednou?

4. *Dijkstra III.: Nechť délky hran leží v množině $\{0, \ldots, L\}$. Navrhněte datovou strukturu založenou na přihrádkách, s níž Dijkstrův algoritmus poběží v čase $\mathcal{O}(nL + m)$. Pokuste se vystačit s pamětí $\mathcal{O}(n + m + L)$.*

Uděláme důležité pozorování. Pokud zastavíme Dijkstrův algoritmus v libovolné chvíli a označíme pomocí z zavřený vrchol s nejvyšším ohodnocením $h(z)$ (neboli poslední zavřený vrchol). Potom pro každý otevřený vrchol o platí $h(o) \in \{h(z) + 1, h(z) + 2, \ldots, h(z) + L\}$.

(Stačí si uvědomit, že jsme vrchol o otevírali nejpozději při zpracování vrcholu z a hodnocení vrcholů nerostou.)

S využitím tohoto pozorování můžeme nahradit haldu v Dijkstrově algoritmu polem $\mathtt{Q}[\mathtt{1}\ldots\mathtt{L}]$, kde políčko $\mathtt{Q}[\mathtt{i}]$ obsahuje spoják vrcholů s ohodnocením právě $h(z)+i$. Navíc každý vrchol má pointer na svou položku ve spojáku (pakliže je otevřený). Rozmyslete si, že pak umíme *DecreaseKey* i *Insert* v konstantním čase a *ExtractMin* v čase $O(L)$.

5. *Rychlíky v Tramtárii: V Tramtárii jezdí po železnici samé rychlíky, které nikde po cestě nestaví. V jízdním řádu je pro každý rychlík uvedeno počáteční a cílové nádraží, čas odjezdu a čas příjezdu. Nyní stojíme v čase t na nádraží a a chceme se co nejrychleji dostat na nádraží b. Navrhněte algoritmus, který najde takové spojení.*

   Proč nemůžeme postavit orientovaný ohodnocený graf a spustit Dijkstrův algoritmus?

   Řešením je stavit graf tzv. za běhu. Na začátku otevřu startovní nádraží s ohodnocením (časem příjezdu) nastaveným na 0. V každém kroku pak vyberu otevřené nádraží s minimálním časem příjezdu pro všechny destinace, kam z něj jedou rychlíky, spočítám jak nejrychleji bych se do nich uměl dostat a případně aktualizuji časy příjezdu pro dané nádraží.

6. *Vícero kritérií: Silnice v mapě máme ohodnocené dvěma čísly: délkou a mýtem (poplatkem za projetí). Jak najít nejlevnější z nejkratších cest?*

   Každou cestu můžeme hodnotit dvojicí (`délka cesty`, `cena za projetí`) a dvě cesty potom porovnáváme lexikograficky - tj. kratší je vždy lepší a pokud jsou stejně dlouhé pak je menší ta levnější. Nyní můžeme použít Dijkstru jako obvykle, jen místo hodnocení používáme tyto dvojice s popsaným uspořádáním. Co kdybychom chtěli nějaký kompromis, kupříkladu by se nám nelíbila nejkratší cesta, která je dvakrát dražší než o kilometr delší levná cesta. To bychom mohli modelovat třeba minimalizováním funkce $\texttt{délka cesty} + 0.5 * \texttt{cena cesty}$ (co by se změnilo?)

7. *Vysoké kamiony: Mějme mapu města ve tvaru orientovaného grafu. Každou hranu ohodnotíme podle toho, jaký nejvyšší kamion po dané ulici může projet. Po cestě tedy projede maximálně tak vysoký náklad, kolik je minimum z ohodnocení jejích hran. Jak pro zadané dva vrcholy najít cestu, po níž projede co nejvyšší náklad?*

   Opět lehce modifikujeme Dijkstru. Hodnocení vrcholů na začátku nastavíme $h(u) = \infty$ pro počáteční vrchol a $h(w) = 0$ pro všechny ostatní. Ke zpracování vybíráme vždy maximální prvek z (tj. místo kam se můžeme dostat s nejvyšším nákladem), a pro každého jeho souseda w spočítáme jestli cestou přes z dokážeme dopravit vyšší náklad než doposud umíme (tj. jestli je $\min(h(z), \texttt{výška hrany } zw)$ větší než aktuální $h(w)$) a případně aktualizujeme $h(w)$.