Reikniritrit Kruskals (1956)

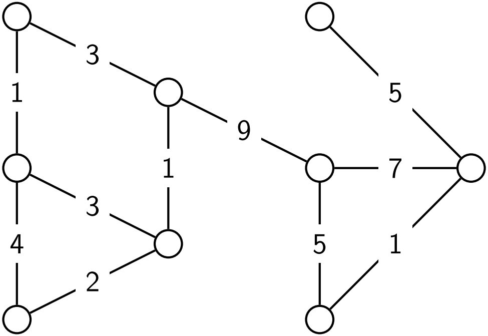
Bergur Snorrason

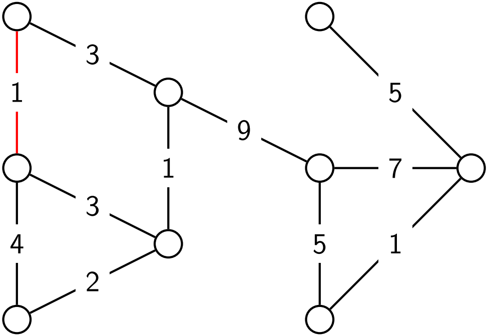
6. mars 2023

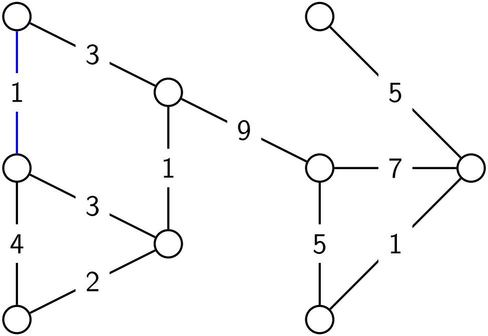
- Gerum ráð fyrir að við séum með samanhangandi óstenft net $G = (V, E)$.
- Munið að net kallast tré ef það er samanhanhandi og órásað.
- Auðvelt er að sýna með þrepun að við getum gert G að tréi með því að fjarlægja leggi.
- Einnig má sýna að tré uppfyllir alltaf $|E| = |V| - 1$.
- Ef G' er tré sem fæst með því að fjarlægja leggi úr G þá köllum við G' *spannandi tré* G (e. *spanning tree*).

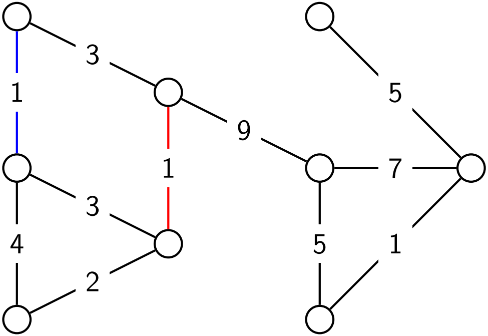
- Ef $G = (V, E, w)$ er vegið net og $G' = (V', E')$ er spannandi tré netsins (V, E) þá segjum við að *stærðin* á spannandi trénu sé
$$S(G') = \sum_{e \in E'} w(e).$$
- Hvernig förum við að því að finna G' þannig að $S(G')$ sé sem minnst.
- Slíkt G' er kallað *minnsta spannandi tré netsins G* (e. *minimum spanning tree*), þó svo að það sé ekki ótvírætt ákvarðað.
- Við getum fundið minnsta spannandi tré gráðugt.

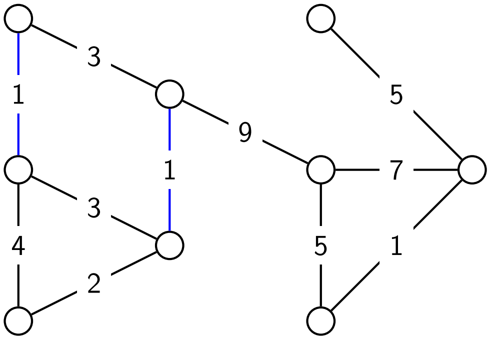
- Við getum lýst aðferðinni í einni málgrein.
- Við bætum alltaf við minnsta leggnum sem myndar ekki rás.
- Hvernig getum við gert þetta á hagkvæman hátt.
- Við byrjum með net með $|V|$ hnúta en enga leggi.
- Gerum ráð fyrir að við höfum bætt við nokkrum leggjum sem mynda ekki rás.
- Ef við viljum bæta við leggnum (u, v) þá erum við í raun að sameina samhengisþættina sem hnútarnir u og v tilheyra.
- Ef þeir tilheyra sama samhengisþætti þá myndast rás við það að bæta við leggnum.
- Svo við getum notað sammengisleit til að segja til um hvort leggur myndi rás.

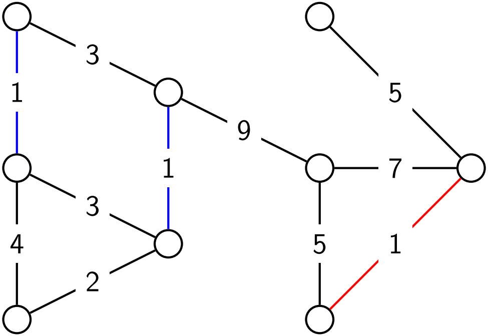

3
1
9
1
3
4
2
5
7
5
1

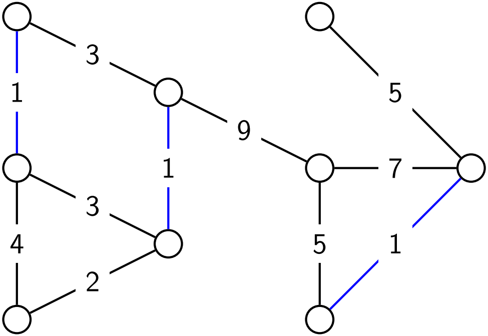

3
1
9
5
1
3
7
4
5
1
2

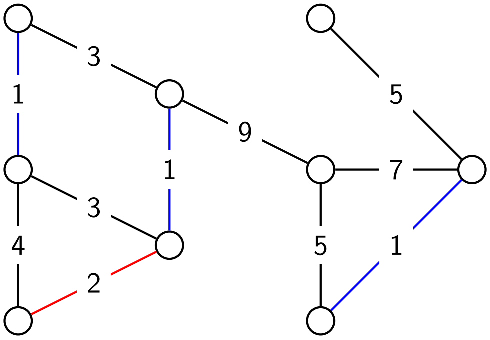

3
1
5
9
1
7
3
4
5
1
2

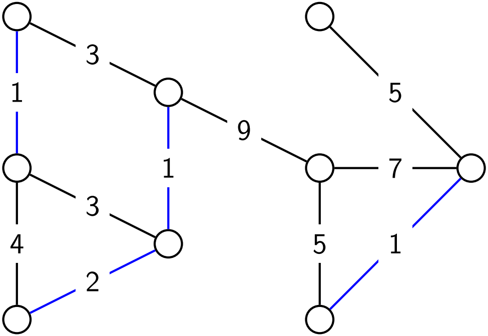

3
1
9
5
1
7
3
4
5
1
2

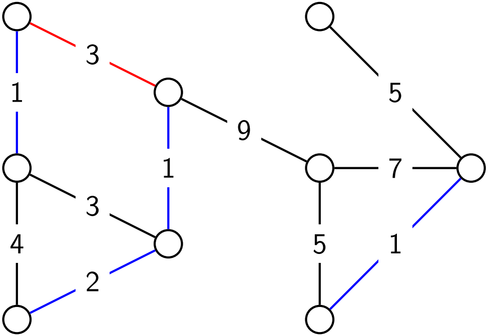

3
1
9
1
3
4
2
5
7
5
1

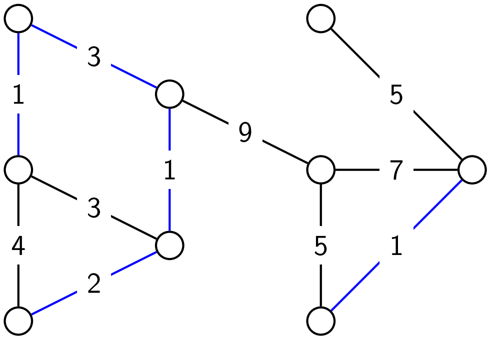

3
1
9
1
3
4
2
5
7
5
1

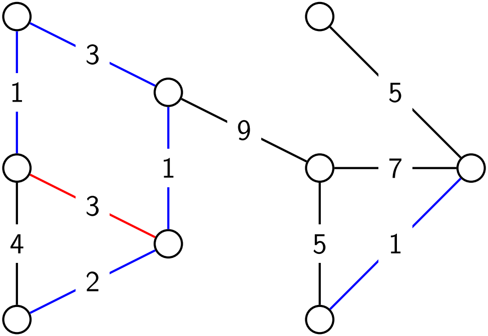

3
1
9
1
3
4
2
5
7
5
1

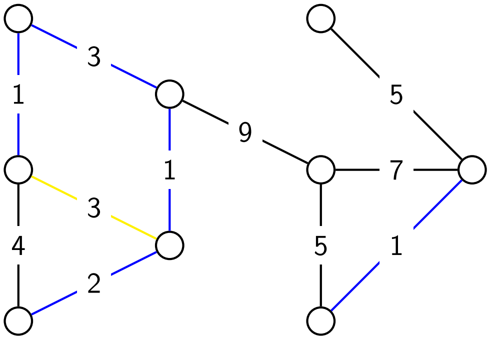

3
1
9
5
7
1
3
4
5
1
2

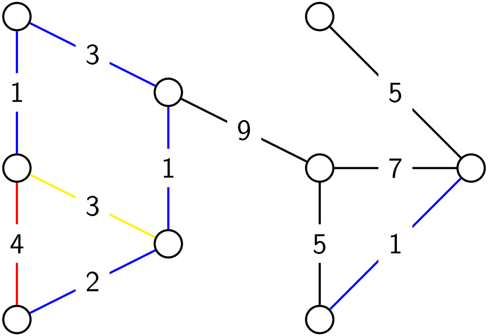

3
1
9
1
3
4
2
5
7
5
1

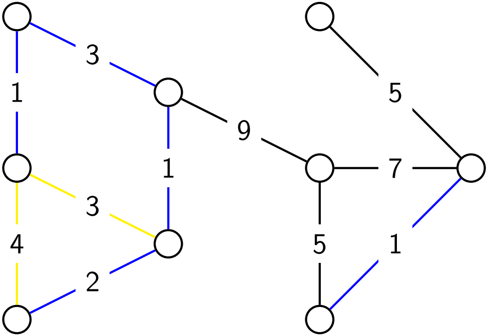

3
1
5
9
1
7
3
4
5
1
2

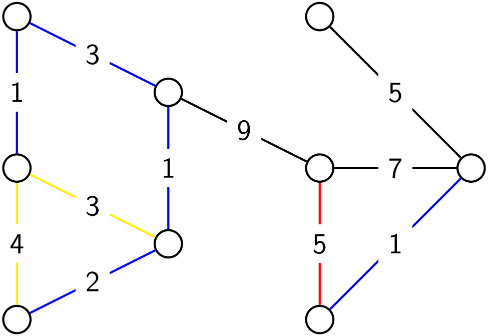

3
1
9
1
3
4
2
5
7
5
1

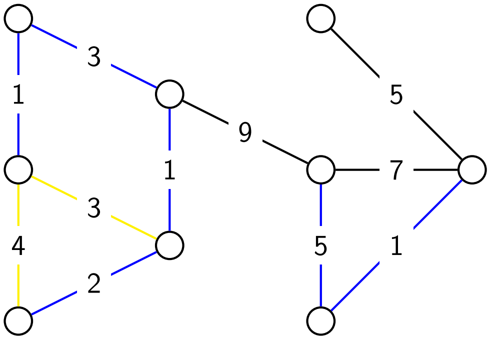

3
1
9
5
1
7
3
4
5
1
2


3
1
9
1
3
4
2
5
7
5
1

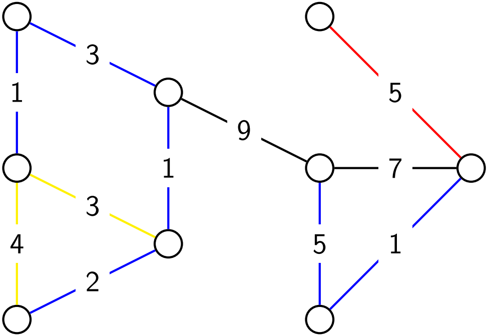

3
1
9
1
3
4
2
5
7
5
1

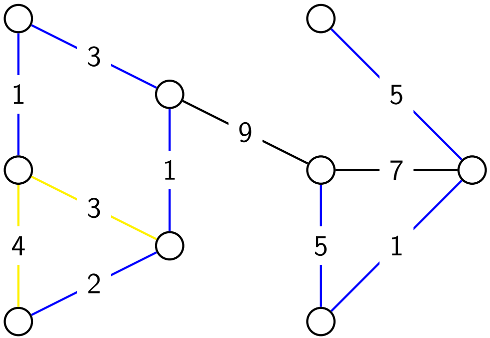

3
1
5
9
1
7
3
4
5
1
2

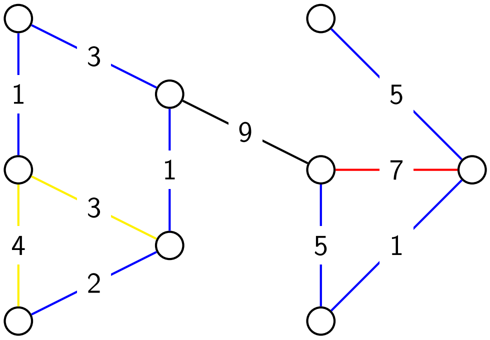

3
1
9
5
1
7
3
4
5
1
2

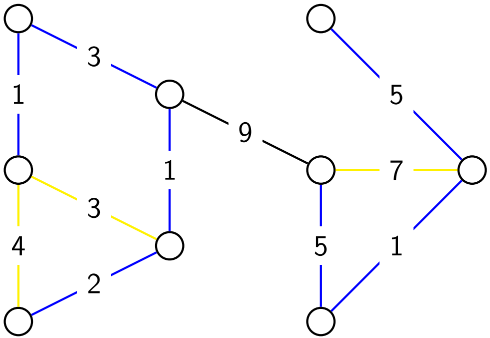


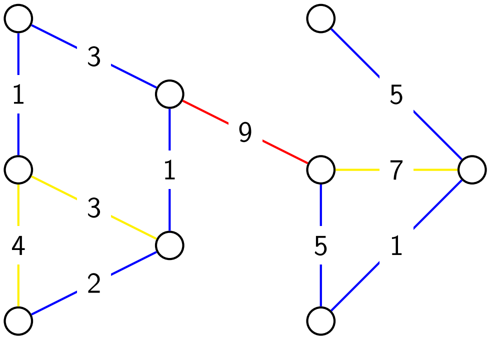
3
1
9
1
3
4
2
5
7
5
1

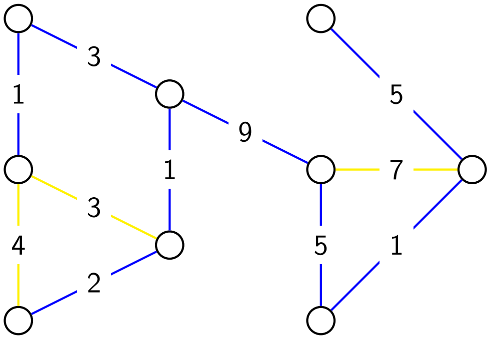

3
1
9
5
1
7
3
4
5
1
2

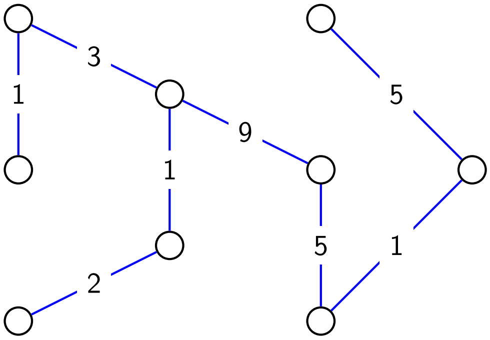

3
1
9
5
1
7
3
4
5
1
2


3
1
9
1
5
7
3
4
2
5
1


3
1
9
5
1
7
3
4
5
1
2


3
1
9
1
3
4
2
5
7
5
1


3
1
9
1
3
4
2
5
7
5
1


3
1
5
9
1
5
1
2

- Við höfum ekki áhuga á nágrönnum nóða heldur vigtum á leggjum svo við notum leggjalista í útfærslunni okkar.
- Við byrjum á að raða leggjalistanum eftir vigt leggjanna.
- Við göngum síðan á leggina og:
  - Gerum ekkert ef leggurinn myndar rás (`find(u) == find(v)`).
  - Bætum leggnum í spannandi tréð ef hann myndar ekki rás og sameinum í sammengisleitinn (`join(u, v)`).

- Þetta reiknirit skilar alltaf spannandi tré, en það er meiri vinna að sýna að það sé ekki til minna spannandi tré.
- Við munum ekki týnast í slíkum smáatriðum hér.

```c
typedef struct { int x, y, z; } ii;
int cmp(const void* p1, const void* p2) { return ((ii*)p1)->z - ((ii*)p2)->z; }

int kruskal(ii* e, ii* t, int n, int m)
{
    int i, j = 0, r = 0, p[n];
    uf_init(p, n);
    qsort(e, m, sizeof(e[0]), cmp);
    for (i = 0; i < m; i++)
    {
        if (uf_find(p, e[i].x) == uf_find(p, e[i].y)) continue;
        r += e[i].z;
        uf_join(p, e[i].x, e[i].y);
        t[j++] = e[i];
    }
    return r;
}
```

- Það fyrsta sem við gerum er að raða leggjunum, sem við gerum í $\mathcal{O}(E \log E)$ tíma.
- Síðan ítrum við í gegn leggina og framkvæmum fastann fjölda af sammengisleitaraðgerðum fyrir hvern legg, sem tekur $\mathcal{O}(E\alpha(V))$ tíma.
- Saman er þetta því $\mathcal{O}(E \log E + E\alpha(V)) = \mathcal{O}(E \log E)$.