4. NPC problems involving sets and numbers

- NPC, Cook's og 3SAT
- $\boldsymbol{3SAT \leq TPM}$
- $\boldsymbol{TPM \leq XC3}$
- $\boldsymbol{XC3 \leq KNAPSACK}$

NP er en anden kompleksitetsklasse, som er formelt defineret ved:

$$\forall x \in \{0,1\}^*: x \in L \Leftrightarrow [\exists y \in \{0,1\}^*: |y| \leq p(|x|) \wedge \langle x, y \rangle \in L'] \quad , \quad L' \in \boldsymbol{P} \quad , \quad p \; er \; poly.$$

Dvs. vi vil kunne tjekke om en given løsning er en løsning i polynomiel tid – ydermere må denne løsning maks. have længden polynomisk af probleminstansen.

Dermed kunne vi afgøre om $x \in L$ for $L \in \boldsymbol{NP}$ ved at lave en exhaustive search, hvor der for alle mulige $2^{p(|x|)+1} - 1$ værdier af y tjekkes om $\langle x, y \rangle \in L'$.

NP-hårde sprog er de sprog, der har ret lille sandsynlighed for at ligge i P. Formelt kan vi sige

$$\forall L' \in \boldsymbol{NP}: L' \leq L \Rightarrow L \in NPH$$

NP-hårde sprog ligger ikke nødvendigvis i NP, men de der gør er ret interessante, idet der er en del naturlige problemer, vi ønsker at løse, hvor vi ved, vi ikke kan finde en effektiv løsning (hvis $P \neq NP$). Problemer i både NP og NPH kaldes **NP-complete.**

Lemma 7: Hvis L_1 er NPH og $L_1 \leq L_2$ så er L_2 NPH. **Bevis:** Transitivitet gælder, og idet alle problemer i NP reducerer til L_1 reducerer disse også til L_2, og def. for NPH er opfyldt.

Lemma 7 giver os en anledning til at finde et problem, som er NPC, hvor vi vha. lemma 7 kan reducere andre mulige NPC-problemer til, således vi ikke behøver fører et langt, akavet bevis, hver gang vi vil vise, at et problem er NPC. Cook's sætning gør nøjagtigt dette.

Cook's sætning giver os et problem SAT, som er NP-hårdt (og ligger i NP). Dermed ligger det i NPC. Vi viser at CircuitSAT er NP-hårdt og reducerer til SAT, hvormed Cook's sætning er vist. CircuitSAT generaliserer SAT, idet CNF'er er formularer, som er kredsløb.

3SAT: Vi er givet en 3CNF-formel Φ, og skal bestemme om vi kan finde en sandhedstilordning T som tilfredsstiller Φ. Da 3SAT er NP-complete (og en specialisering af SAT, idet vi reducerer herfra), kan vi bruge 3SAT til at vise, at andre problemer er NPC.

TRIPARTITE-MATCHING: Givet tre mængder B, G, H med hver n elementer, og relationen $T \subseteq B \times G \times H$, kan vi finde n tripler i T, hvor ingen af dem har noget til fælles?

$\boldsymbol{3SAT \leq TRIPARTITE\ MATCHING}$:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$$

Vi skal konstruere en slags graf ud fra 3SAT-formlen. Vi skal lave en choice-constraint-gadget for hver variabel. Denne skal indeholde $2k$ H, hvor $k = \max(|x_i|, |\neg x_i|)$. Her er der i alt $2k$ tripler.

Herudover er der k G og k B. Ydermere er der et B-G-par for hver klausul. Dette par indgår i i alt 3 tripler (en for hver literal pr. klausul). Det hjem, der er med i det par, der er med i matchingen her, angiver sandhedsværdien for literalen (og dermed for variablen).

Til sidst tilføjes der et antal B-G-par, så der er lige mange B-G-par, som der er H i alt. Der vil altid være mindst 3 gange så mange H som klausuler.

Der findes en matching ⇔ 3SAT-formlen opfyldes af en sandhedstilordning.

Bevis: Hjem valgt i blomsterne er af "ens" type – dvs. der kan ikke både vælges x og $\neg x$.

EXACT-COVER-BY-3-SETS: Vi er givet en familie $F = \{S_1, \dots, S_n\}$ af undermængder af U, således at $|U| = 3m$, for et tal m og $|S_i| = 3$. Er der m mængder i F som er disjunkte (adskilt) og tilsammen danner U?

$XC3 \in NPC$: XC3 er en generalisering af TPM: TPM er tilfældet, hvor $U = B \cup G \cup H = \{1, \dots, 3m\}$. Til enhver valgt triple $t_i = (b, g, h) \in T$ fra TPM har vi $S_i = (b, g, h)$ i XC3, for $i = 1, \dots, n$. Hermed er foreningsmængden af de m tripler kun er U, hvis der er en matching i TPM og omvendt.

KNAPSACK er et integer programming tilfælde. Vi har n items: $\langle v_i, w_i \rangle$. Vi skal finde en delmængde $S \subseteq \{1, \dots, n\}$, hvor der gælder at$\sum_{i \in S} w_i \leq W$ og for et tal K: $\sum_{i \in S} v_i \geq K$.

$XC3 \leq KNAPSACK$:

Vi kigger på specialtilfældet af knapsack, hvor $v_i = w_i$ og $K = W$. Vi er givet n heltal $w_1, \dots, w_n$ og ønsker at se om delmængde af S summer op til præcis K. Dette kaldes og SUBSET-SUM:

Hvert S_i kan vi se som et item i knapsack – vi oversætter til $\{0,1\}^{3m}$. Disse sættes ind i en tabel, således, at hver række repræsenterer et item (kolonner er elementer i XC3). Nu handler det helt simpelt om at addere disse rækker, således vi får en række med kun 1'ere $= K = 2^{3m} - 1$. Så vælges de items ud, som præcis adderer op til denne række.

Dvs. hvis vi har et XC3, som opfyldes, så har vi også instans af knapsack, som opfyldes. Dette gælder også den anden vej.

Vi skal dog passe på binært addering, idet følgende er forkert:

0011	{3,4}
0101	{2,4}
0111	{2,3,4}
1111	

Da vi har mængde i addition, skal vi altså sikre, at vi ikke får et forkert resultat. Vi tænker på vores items som items i base $n + 1$ i stedet for 2. Dermed vil det gå op, idet vi i ovenstående tilfælde vil få 0223 i sidste række, hvilket ikke er 1111.