1. In het bewijs van de stelling van Cook–Levin zetten we een input w voor een gegeven probleem in NP om in een Booleaanse formule. Leg uit welke variabelen precies voorkomen in deze formule, en wat hun interpretatie is. Beschrijf dan wat de formule allemaal uitdrukt over deze variabelen.

Antwoord: Zie boek!

2.

(a) Beschouw de complexiteitsklasse $\text{PSPACE} = \bigcup_k \text{SPACE}(n^k)$. Leg uit waarom $\text{NP} \subseteq \text{PSPACE}$.

Antwoord: Als $L \in \text{NP}$ is er een niet-deterministische Turing machine N die L beslist in polynomiale tijd. We hebben een algemene manier gezien om een niet-deterministische machine te simuleren met een deterministische machine. Uit de manier hoe deze simulatie werkt is duidelijk dat als de niet-deterministische slechts polynomiaal lange mogelijke runs heeft, dat dan de space die de simulerende deterministische machine nodig heeft ook slechts polynomiaal is.

(b) We hebben gezien dat elk probleem in NL efficiënt gereduceerd kan worden naar *PATH*. Maar waarom volgt daaruit dat $\text{NL} \subseteq \text{P}$?

Antwoord: Omdat $\mathit{PATH} \in \text{P}$.

(c) Beschouw nu de complexiteitsklasse $\text{EXPTIME} = \bigcup_k \text{TIME}(2^{n^k})$. Leg uit waarom $\text{NP} \subseteq \text{EXPTIME}$.

Antwoord: We hebben gezien dat een niet-deterministische machine die in tijd $O(t(n))$ werkt kan gesimuleerd worden door een deterministische machine die in tijd $2^{O(t(n))}$ werkt. Als $t(n) = O(n^k)$ is dus $2^{O(t(n))} = 2^{O(n^k)}$.

(d) We kunnen echter iets veel sterker inzien, namelijk, dat EXPTIME niet slechts NP, maar heel PSPACE omvat! Gebruik een gelijksoortige redenering als in het bewijs dat $\text{NL} \subseteq \text{P}$ (de reachability method) om aan te tonen dat inderdaad $\text{PSPACE} \subseteq \text{EXPTIME}$.

Antwoord: In dat bewijs reduceerden we een willekeurig probleem in NL naar $PATH$. Deze reductie was efficiënt: we stelden de graaf op van alle mogelijk toestanden van een gegeven machine die slechts logaritmisch veel space gebruikt, en we rekenden uit dat deze graaf dan slechts polynomiaal groot is. Een machine die polynomiaal veel space kan gebruiken, heeft echter exponentiëel veel toestanden. Dezelfde reductie is dus niet meer efficiënt, maar is ook niet erger dan exponentiëel. We passen dan $PATH$ toe op een exponentiëel grote graaf, en omdat $PATH$ zelf kan opgelost worden in polynomiale tijd, blijft de tijd dus exponentiëel, zoals moest bewezen worden.

3.

(a) Een *Hamiltonian cycle* in een gerichte graaf G is een pad in G dat vertrekt in een knoop s, en ook terug eindigt in s, zodat alle overige knopen precies 1 keer voorkomen op het pad. Beschouw nu het beslissingsprobleem $HAMCYCLE$: gegeven $\langle G, s\rangle$, is er een Hamiltonian cycle in G vertrekkend in s? Toon de NP-compleetheid aan van $HAMCYCLE$.

Antwoord: In NP: Het certificaat is een Hamiltonian cycle.

NP-compleet: We reduceren $HAMPATH$ naar $HAMCYCLE$. Daartoe moeten we een input $\langle G, s, t\rangle$ omvormen naar een input $\langle H, s'\rangle$ zodat $\langle G, s, t\rangle \in HAMPATH$ als en slechts als $\langle H, s'\rangle \in HAMCYCLE$.

We doen dit als volgt. We stellen H gelijk aan de graaf verkregen door in G een pijl toe te voegen van t naar s (moest die er nog niet zijn), en alle andere pijlen naar s te verwijderen (moesten er zo zijn). We stellen verder s' gelijk aan s. Deze reductie is duidelijk efficiënt te programmeren.

Nu nog correctheid aantonen. Als er een Hamiltonian pad is van s naar t in G, dan kunnen we de toegevoegde pijl van t terug naar s in H gebruiken om een Hamiltonian cycle vanuit s krijgen in H door het Hamiltonian pad naar t te volgen en dan terug in s uit te komen met de toegevoegde pijl van t naar s.

Omgekeerd, als er een Hamiltonian cycle is vanuit s in H, dan kan de laatste knoop in deze cycle, die dus terug naar s gaat, niet anders dan t zijn, want alle andere pijlen naar s hebben we verwijderd. We hebben dus een Hamiltonian pad van s naar t in G.

(b) In het befaamde Traveling Salesman probleem beschouwen we grafen waar de bogen allemaal een *lengte* hebben, waar we natuurlijke getallen voor nemen. Het probleem is dan als volgt: gegeven $\langle G, s, b\rangle$, waar b een bijkomend getal is, is er een Hamiltonian cycle in G vertrekkend in s, van totale lengte hoogstens b? Toon de NP-compleetheid aan van het Traveling Salesman probleem.

Antwoord: In NP: Het certificaat is een Hamiltonian cycle van totale lengte hoogstens b.

NP-compleet: We reduceren *HAMCYCLE* naar *TSP*. Een input $\langle G, s\rangle$ voor *HAMCYCLE* vormen we om naar een input $\langle H, s', b\rangle$ voor *TSP* als volgt. Als H nemen we G, waar we aan elke edge lengte 0 toekennen. Als s' nemen we s. Als b nemen we 0.

Als er een Hamiltonian cycle in G is, is er dus ook een in H, en omdat alle edges lengte 0 hebben, is de totale lengte ook 0. Omgekeerd, als er een Hamiltonian cycle in H is, is er ook een in G, want H is gelijk aan G.

4. Toon de NP-compleetheid aan van het volgende beslissingsprobleem *SET-COVER*. Gegeven $\langle U, S_1, \ldots, S_m, k\rangle$, waarbij U een eindige verzameling getallen is, elke S_i een deel van U, en $k \leq m$ een natuurlijk getal. Kunnen we uit de S_i's er k uitkiezen zodat hun unie reeds volledig U is?

Antwoord: In NP: Het certificaat is een lijst van k indices uit $\{1, \ldots, m\}$ zodat de unie van alle S_i's met i uit deze lijst, heel U geeft.

NP-compleet: We reduceren *VERTEX-COVER* naar *SET-COVER*. Een input $\langle G, k\rangle$ voor *VERTEX-COVER* vormen we om naar een input $\langle U, S_1, \ldots, S_m, k'\rangle$ voor *SET-COVER* als volgt. We nummeren de edges van G. Als U nemen we dan de verzameling van deze nummers. We nummeren ook de knopen van G. Als m nemen we het aantal knopen in G. Voor knoop nummer i stellen we S_i gelijk aan de verzameling van de nummers van alle edges incident aan knoop i. Als k' nemen we k.

Als er nu k knopen bestaan in G zodat elke edge incident is met 1 van deze k knopen, zal de unie van de k S_i's overeenkomend met deze knopen bestaan uit alle edges, dus heel U. Omgekeerd, als er k S_i's bestaat zodat hun unie heel U is, betekent dit dat elke edge incident is met tenminste een knoop overeenkomend met een van deze k S_i's. Deze k knopen vormen dus een vertex cover voor G.

5. Geef een efficiënte reductie van *3SAT* naar het probleem van Integer Linear Programming, dat gedefiniëerd is als volgt. Gegeven een stelsel ongelijkheden tussen lineaire veeltermen over meerdere veranderlijken, met gehele coëfficiënten. Bestaat er een oplossing bestaande uit gehele getallen?

Voorbeelden van zulke stelsels zijn

$$\begin{aligned} x &\leq 2 \\ y &\leq 2 \\ x + y &\geq 6 \end{aligned}$$

(dit heeft geen oplossing) en

$$\begin{aligned} x + y - 3z &\geq 0 \\ z &\geq 10 \end{aligned}$$

(dit heeft wel een gehele oplossing, b.v., $x = 20$, $y = 20$, $z = 10$).

Antwoord: We moeten een Booleaanse formule ϕ in 3cnf omvormen tot een stelsel lineaire ongelijkheden, zodat ϕ satisfiable is als en slechts als het stelsel een gehele oplossing heeft.

Voor elke variabele x uit ϕ nemen we twee veranderlijken x en $\bar{x}$ in het stelsel, en we voegen alvast de gelijkheid $x + \bar{x} = 1$ toe (eigenlijk de twee ongelijkheden $x + \bar{x} \leq 1$ en $x + \bar{x} \geq 1$. Dit garandeert dat eender welke oplossing ofwel x op 1 stelt en $\bar{x}$ op 0, of omgekeerd.

Voor elke clause $a \vee b \vee c$ uit ϕ voegen we nu bijkomend de ongelijkheid $a + b + c \geq 1$ toe. Dit besluit de constructie van het stelsel, die duidelijk efficiënt te programmeren is.

Als nu ϕ satisfiable is, is er een truth assignment α die elke variabele afbeeldt op 0 of 1, zodat elke clause in ϕ voldaan is. Deze truth assignment geeft ons een oplossing voor het stelsel (waar we voor elke variabele, $\alpha(\bar{x})$ definiëren als "**if** $\alpha(x) = 1$ **then** 0 **else** 1"). Inderdaad, elke gelijkheid $x + \bar{x} = 1$ is voldaan, en ook elke ongelijkheid $a + b + c \geq 1$ is voldaan omdat α elke clause voldoet, dus tenminste a, b, of c is 1. Dus het stelsel heeft een gehele oplossing via α.

Omgekeerd, als het stelsel een gehele oplossing heeft, weten we door de gelijkheden $x + \bar{x} = 1$ al zeker dat voor elke variabele x, de overeenkomstige veranderlijke x in de oplossing ofwel 1 ofwel 0 is, en de veranderlijke $\bar{x}$ juist het tegengestelde. Onze oplossing geeft ons dus een truth assignment op de variabelen. Verder, omdat elke ongelijkheid $a + b + c \geq 1$ voldaan is, zal deze truth assignment elke clause voldaan. Dus ϕ is satisfiable door deze truth assignment.