

Hamiltonian Circuit Problem



Das Hamiltonian Circuit Problem ist NP-vollständig, und zwar schon falls der Graph planar und 3-regulär ist.

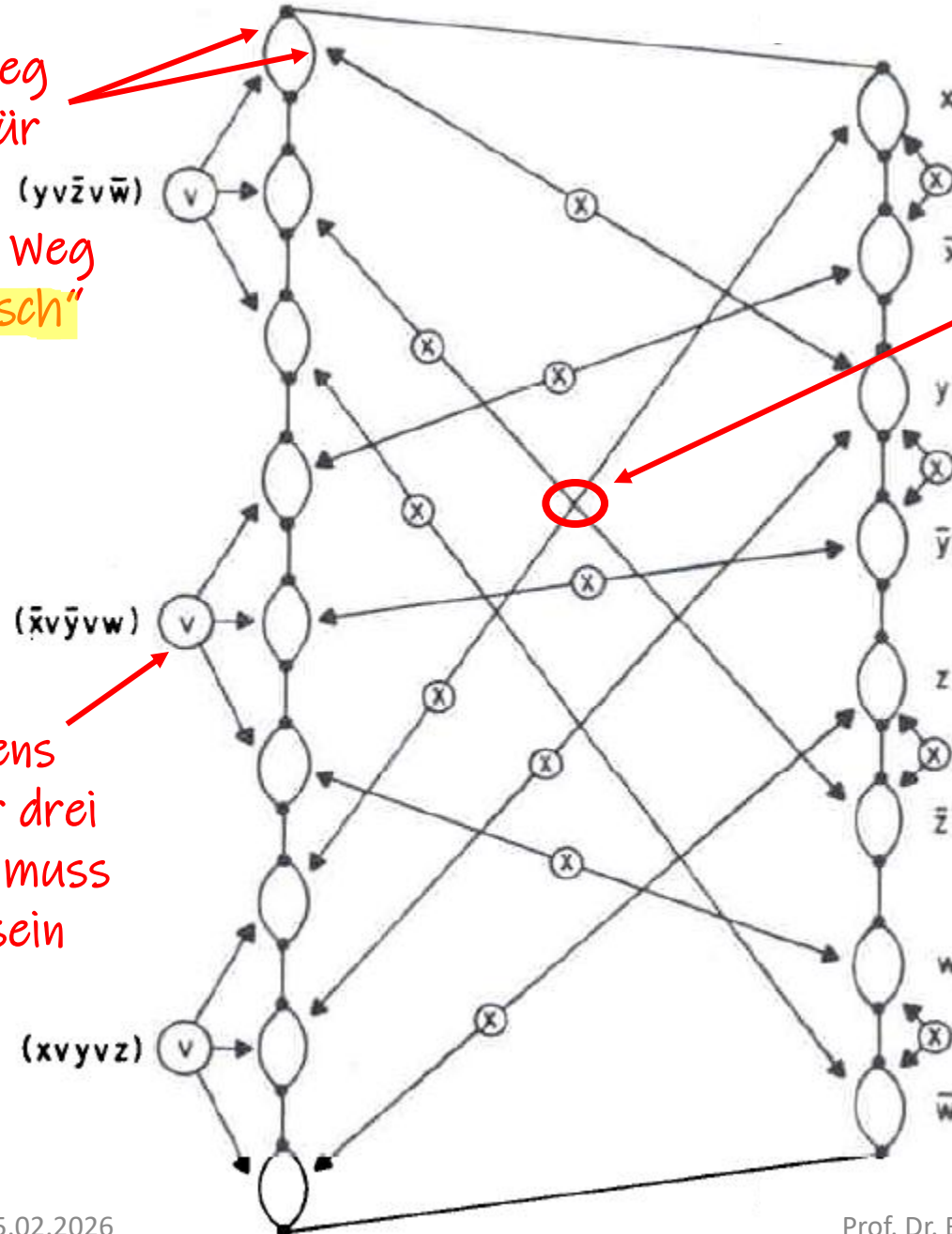
Graph, welcher in der Ebene dargestellt werden kann, so dass sich keine zwei Kanten schneiden

jeder Knoten besitzt genau 3 Nachbarn

für einen (ungerichteten) Graph soll entschieden werden, ob ein Rundweg existiert, welcher jeden Knoten genau einmal enthält



linker Weg
steht für
„wahr“, $(yv\bar{z}v\bar{w})$
rechter Weg
für „falsch“



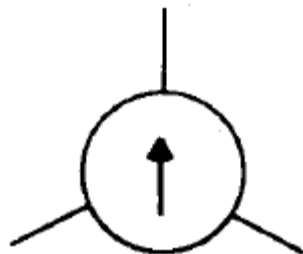
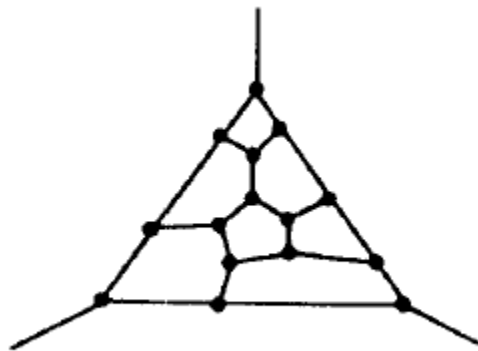
„Kreuzung“: Kanten des Graphen
dürfen sich nicht kreuzen

Wir zeigen die Konstruktion an
einem Beispiel: gesucht ist ein
Graph (mit den genannten
Eigenschaften) welcher genau dann
einen (Hamilton-) Rundweg besitzt,
falls die unten stehende Formel
erfüllbar ist.

$$F = (x \vee y \vee z) \wedge (\bar{x} \vee \bar{y} \vee w) \wedge (y \vee \bar{z} \vee \bar{w})$$

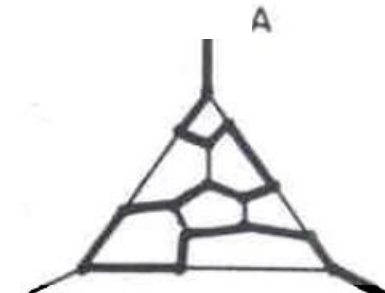
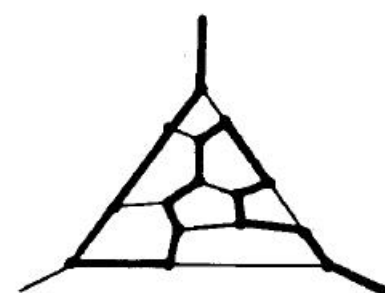
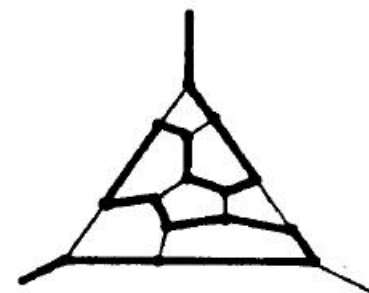
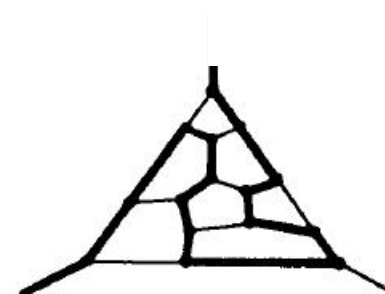
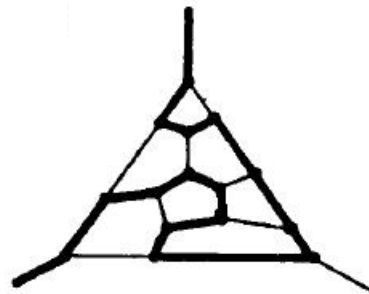
„xor“: genau einer der
beiden Wege gehört
zum Pfad

Tutte Fragment

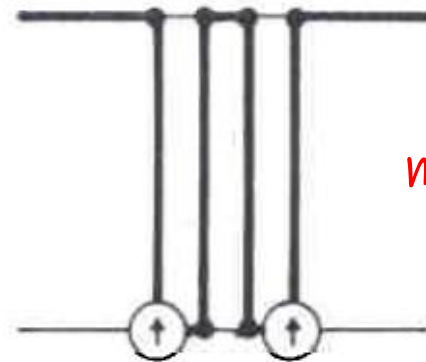
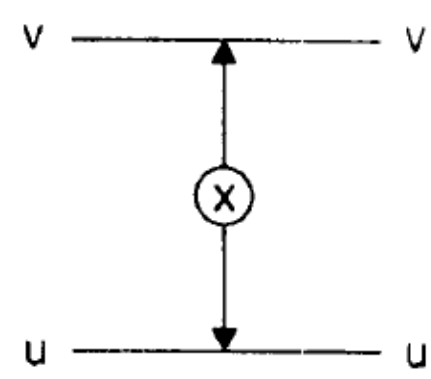
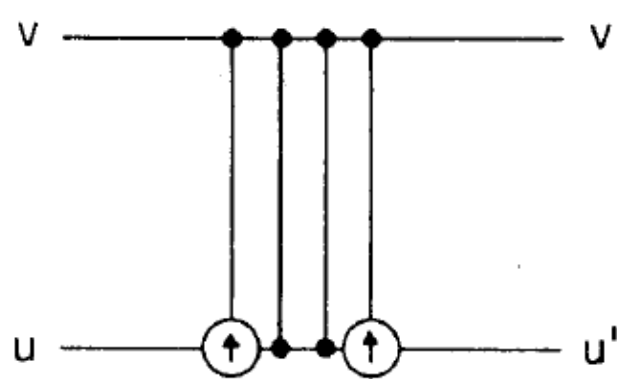


Symbol

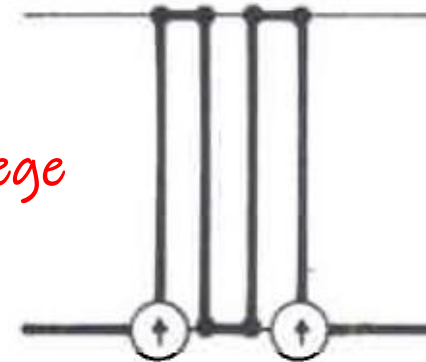
mögliche Wege



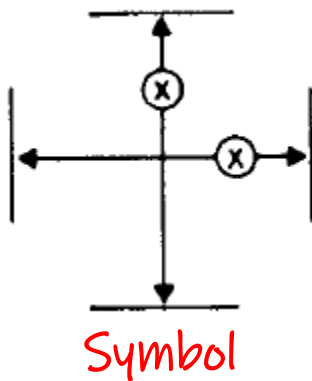
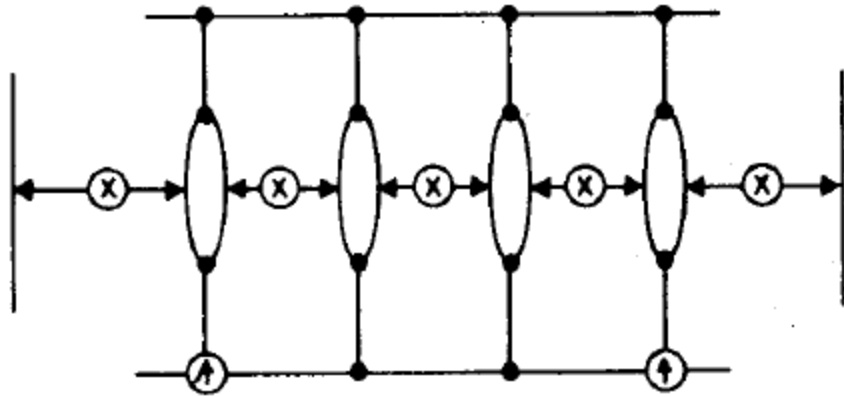
„Exclusive-OR“



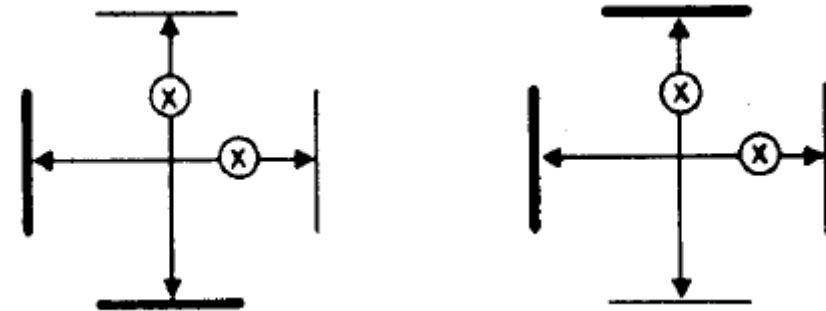
mögliche Wege



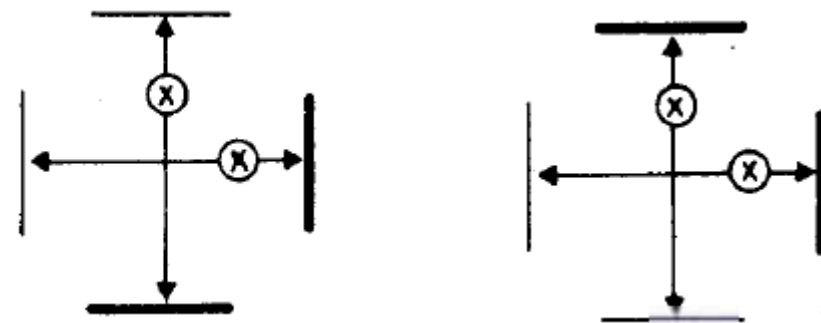
„Kreuzung“



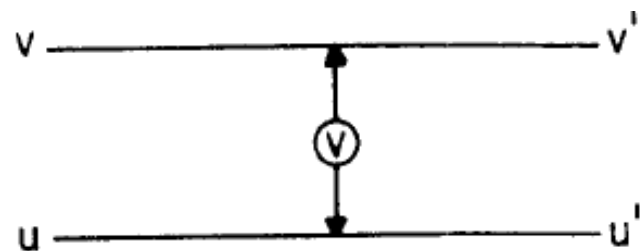
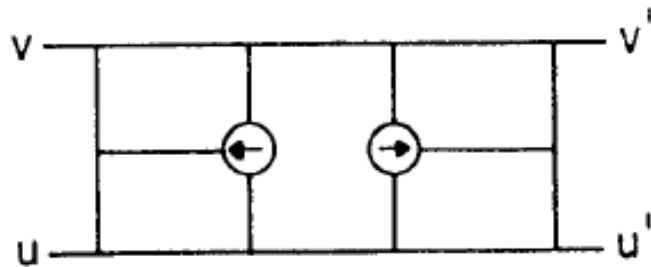
Symbol



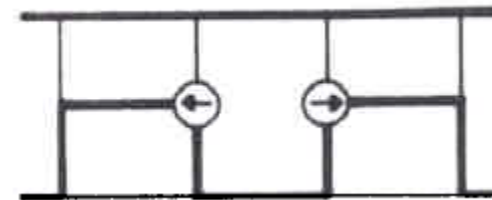
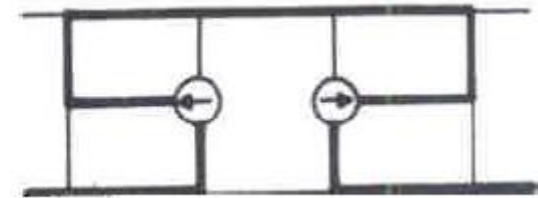
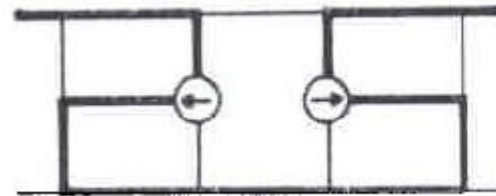
mögliche Wege



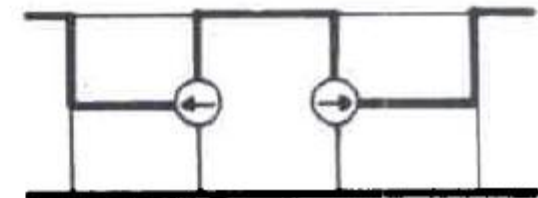
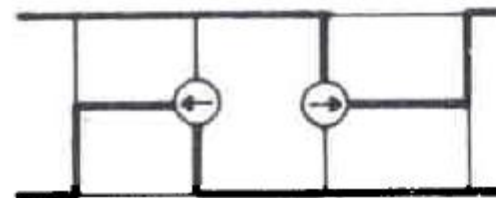
„OR“



Symbol

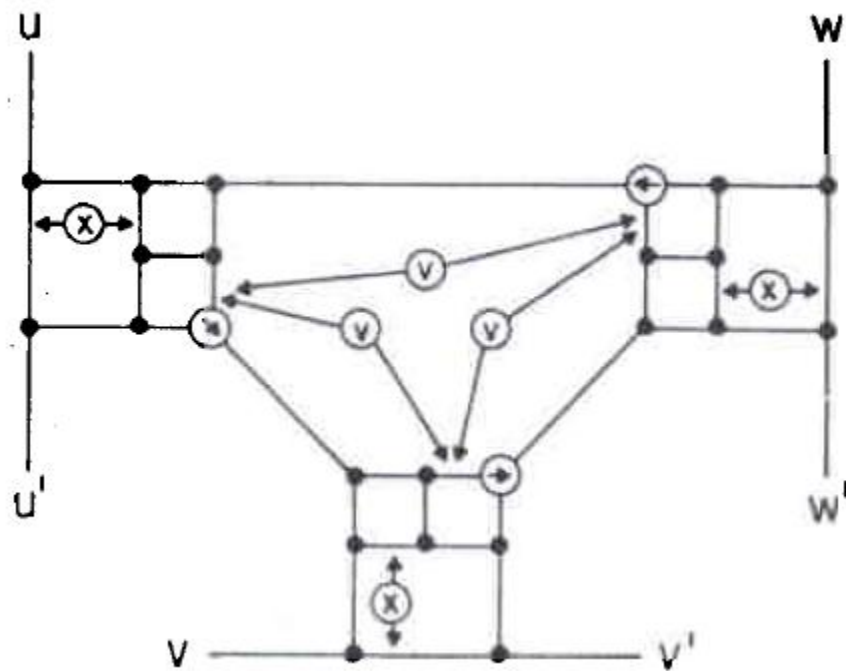


mögliche Wege

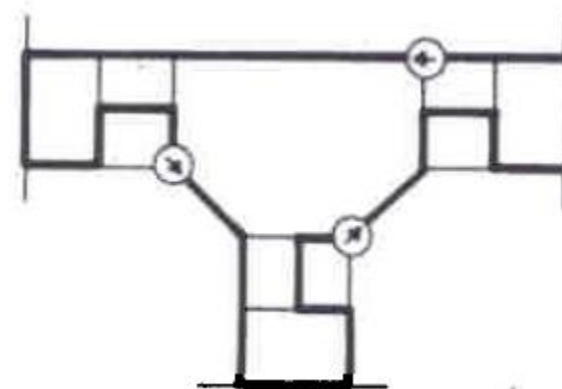
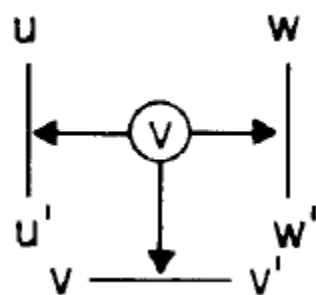




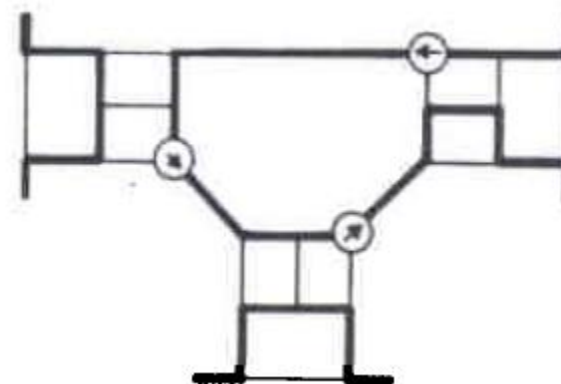
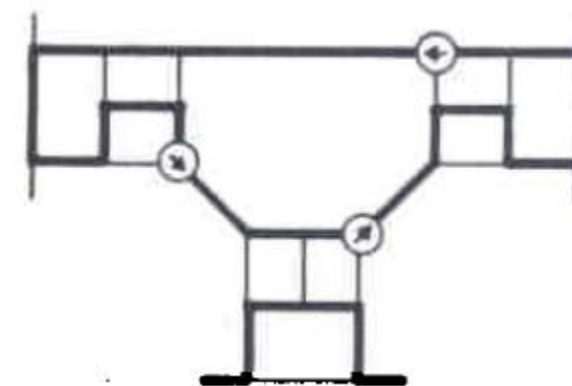
„(trippel-)OR“



Symbol



mögliche Wege
(ohne Symmetrie)



MINESWEEPER



Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law



MINESWEEPER



Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

	2	2	2	2	
	2	0	0	2	
	2	0	0	2	
	2	2	2	2	

MINESWEEPER ist NP-vollständig

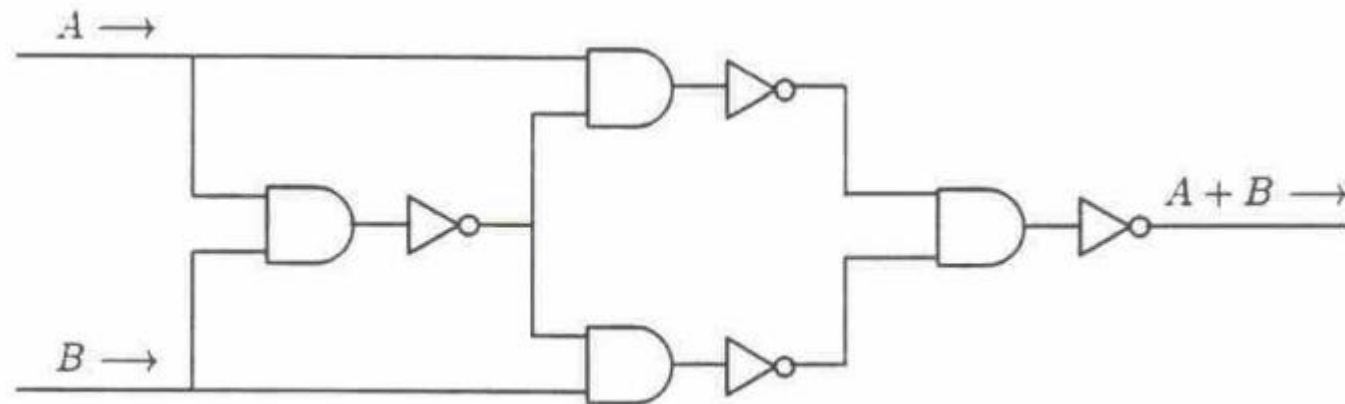


Figure 12. Making an XOR gate with AND and NOT gates.

MINESWEEPER ist NP-vollständig



$X \rightarrow$

...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
...	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	...
...	x	1	x'	x	1	x'	x	1	x'	x	1	x'	x	1	x'	x	...
...	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	...
...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...

Figure 6. A wire.

MINESWEEPER ist NP-vollständig

$X \rightarrow$				1	2	2	1	
...	1	1	1	2	*	*	3	1
...	1	x'	x	1	x'	*	*	2
...	1	1	1	1	2	x	*	2
					1	1	2	1
					1	x'	1	
					1	x	1	X
					1	1	1	\downarrow
					\vdots	\vdots	\vdots	

1	1	1	X \rightarrow					
2	*	3	1	1	1	1	1	...
3	*	x'	x	1	x'	x	1	...
2	*	3	1	1	1	1	1	...
1	1	1						

				:	:	:				
				1	x	1				
				1	x'	1				
X →				1	1	1			X' →	
..	1	1	1	1	x	1	1	1	1	..
..	x'	x	1	x'	2	x'	1	x	x'	..
..	1	1	1	1	x	1	1	1	1	..
				1	1	1				
				1	x'	1				
				1	x	1				
				:	:	:				

MINESWEEPER ist NP-vollständig

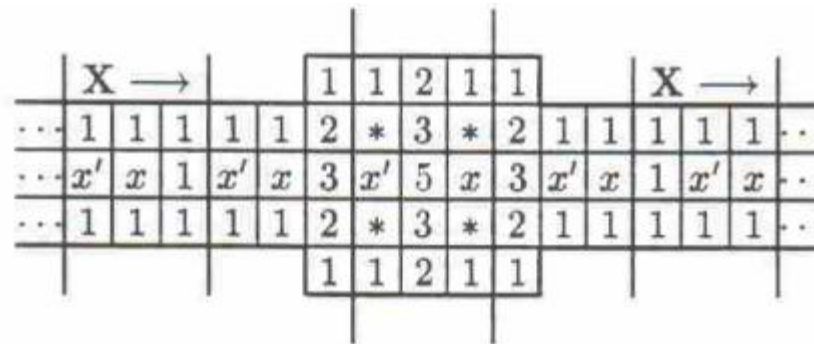


Figure 10. A phase-changer made from two NOT gates.

MINESWEEPER ist NP-vollständig

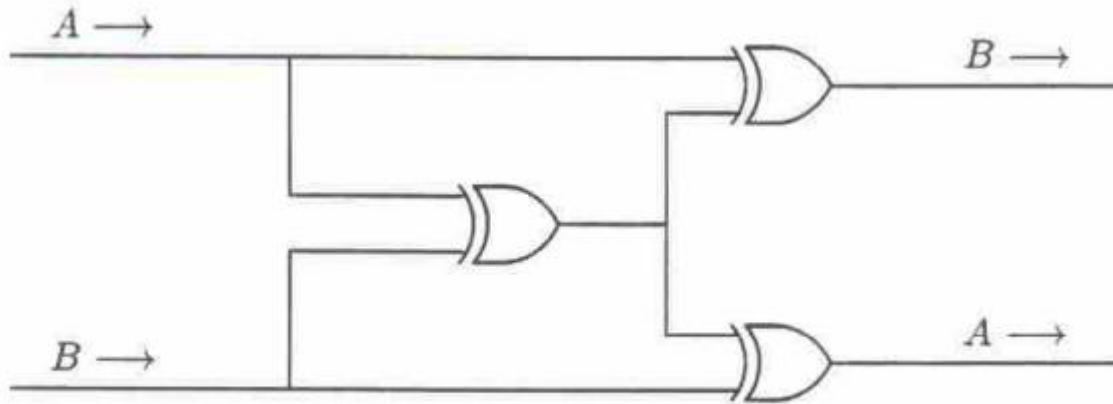


Figure 11. Crossing two wires with three XOR gates.

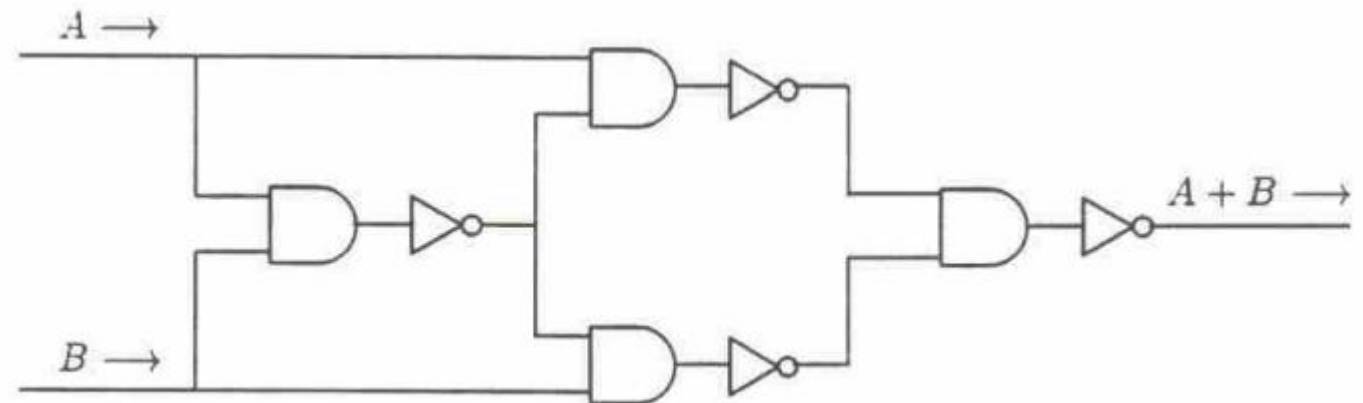


Figure 12. Making an XOR gate with AND and NOT gates.

MINESWEEPER ist NP-vollständig



$X \rightarrow$									$X' \rightarrow$					
...	1	1	1	1	1	2	*	2	1	1	1	1	1	...
...	x'	x	1	x'	x	3	x'	3	x	x'	1	x	x'	...
...	1	1	1	1	1	2	*	2	1	1	1	1	1	...
						1	1	1						

Figure 9. A NOT gate.

MINESWEEPER ist NP-vollständig

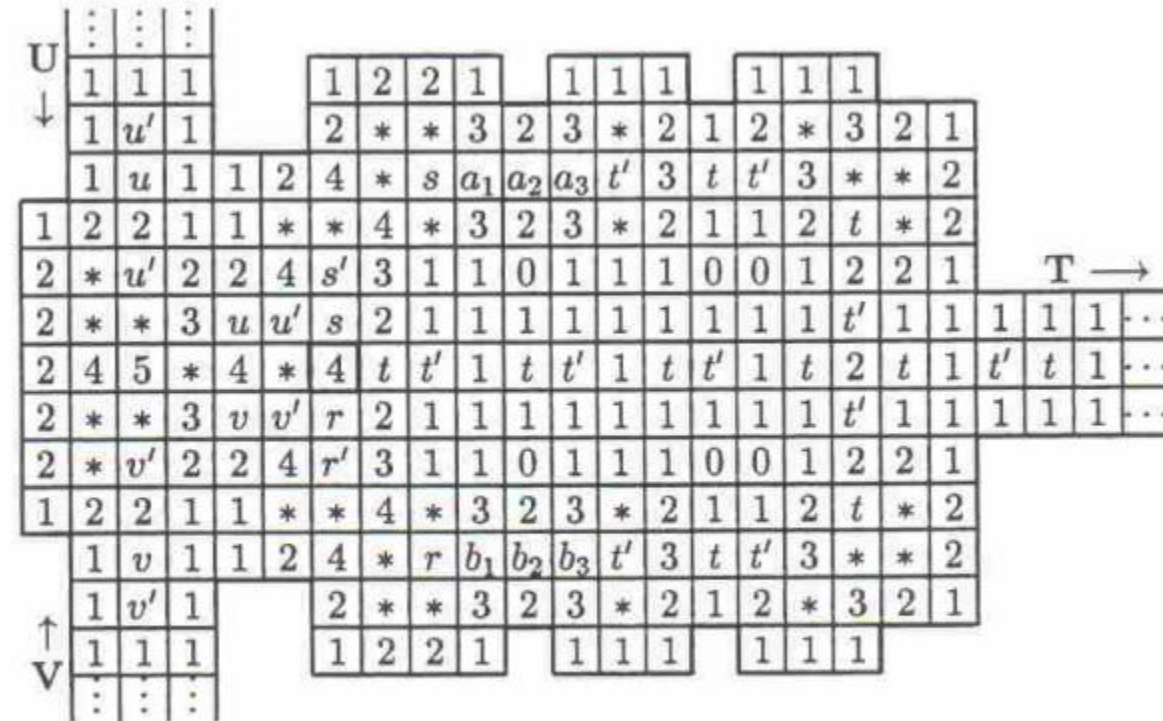


Figure 13. An AND gate.

„Gaming is hard“, Part I



Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

↙ d.h. bestimmte Punkte
müssen erreicht werden



(a) Initial configuration.



(b) Configuration after a traversal from left to right.



(c) On the second traversal attempt, a boulder blocks the way.

Metatheorem 1. Any game exhibiting both *location traversal* (with or without a starting location or an exit location) and *single-use paths* is NP-hard.

Wir bauen einen Level, welcher einen *planaren, 3-regulären Graph* darstellt, wobei die Knoten die Punkte sind, welche besucht werden müssen und *jede Kante ein „single-use-path“* darstellt. Nach Wahl eines Startpunktes (Knoten) „S“ verbinden wir diesen mit einem weiteren (End-)Punkt, welcher nur eine Kante zum Startknoten hat und ansonsten keine weiteren Nachbarn besitzt. Der Level ist nun genau dann *lösbar*, falls ein *Hamilton-Kreis* von „S“ nach „S“ existiert.

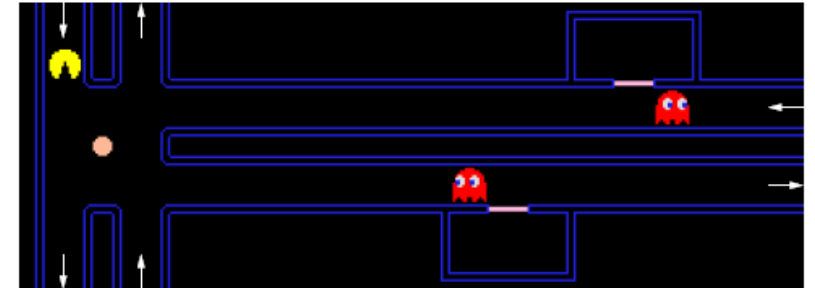
„Gaming is hard“, Part I

Metatheorem 2. *A game is NP-hard if either of the following holds:*

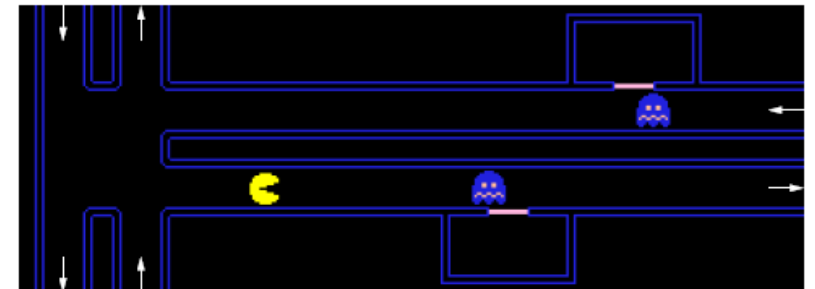
- (a) *The game features collectible tokens, toll roads, and location traversal.*
- (b) *The game features cumulative tokens, toll roads, and location traversal.*
- (c) *The game features collectible cumulative tokens, toll roads, and the avatar has to reach an exit location.*

um diesen Weg zu passieren,
muss ein Token eingesammelt
werden

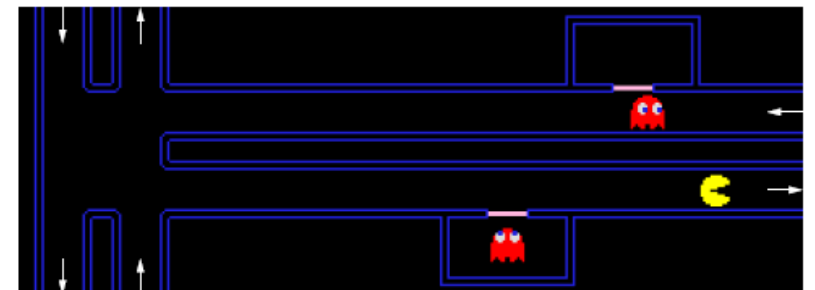
(a) und (b) wie bei MT 1: „tokens“ und „toll roads“ ergeben „single-use-paths“. Bei (b) hat der Spieler zu begin $n+1$ „tokens“. Bei (c) plazieren wir 2 „tokens“ an jeden Punkt und jede Kante wird eine „toll road“, außer der Kante zwischen Start- und Endpunkt, welche eine Sequenz von n „toll roads“ ist.



(a) Initial configuration, approached from the top.



(b) Collecting the power pill to traverse the corridor.



(c) Exiting to the right while the ghosts recover.

„Gaming is hard“, Part I



eine Tür wird von einem Schlüssel
geöffnet, welcher danach als
verbraucht gilt



Metatheorem 3. *A game is NP-hard if it contains **doors** and one-way paths, and either of the following holds:*

- (a) *The game features **collectible keys** and location traversal.*
- (b) *The game features **cumulative keys** and location traversal.*
- (c) *The game features collectible cumulative keys and the avatar has to reach an exit location.*

(alles genau wie bei MT 2: „keys“ = „tokens“, „doors“ = „toll roads“)

„Gaming is hard“, Part I

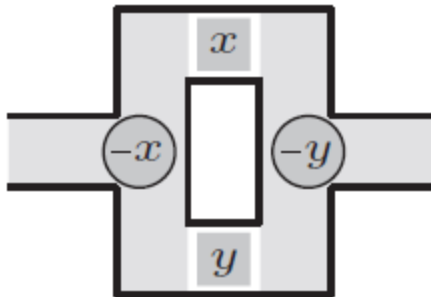


es gibt sowohl Druckplatten, welche
Türen öffnen, als auch schließen



Metatheorem 4. *If a game features **doors** and **pressure plates**, and the avatar has to reach an exit location in order to win, then:*

*Even if no two pressure plates control the same door, the game is **NP-hard**.*



Alles wie bei MT 1: damit alle Punkte besucht werden
müssen, sind vor dem Ausgang n Türen, welche mit
Druckplatten bei jedem Punkt geöffnet werden
müssen

„single-use-path“ mit Druckplatten

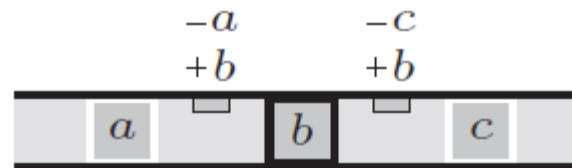
„Gaming is hard“, Part I



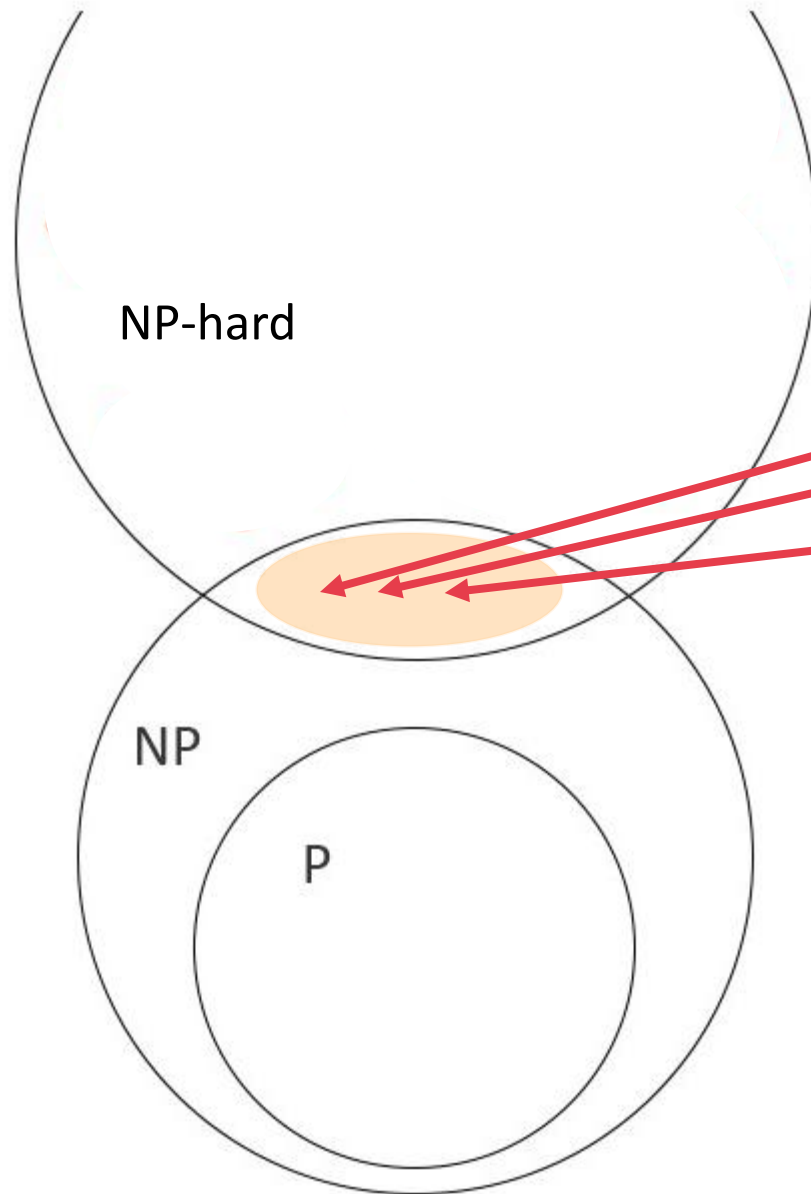
Knöpfe sind wie Druckplatten, außer dass der Spieler entscheiden kann, ob er ihn drücken will.
Ein k -Button beeinflusst k Türen

Metatheorem 5. *If a game features **doors** and **k -buttons**, and the avatar has to reach an exit location in order to win, then:*

*If $k \geq 2$, then the game is **NP-hard**.*



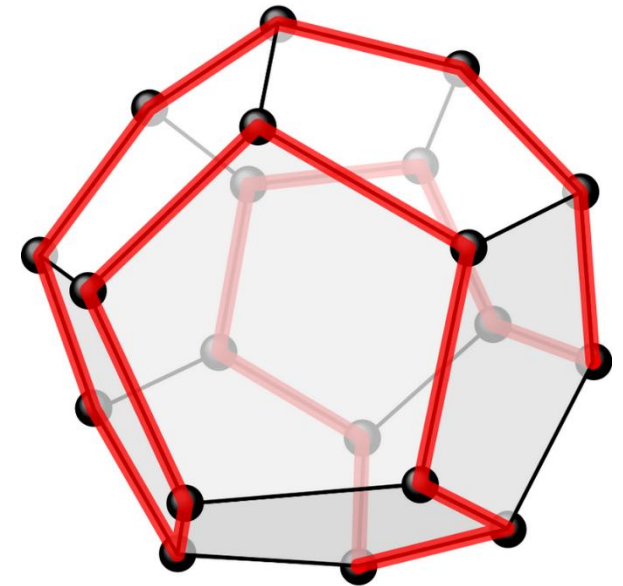
„single-use-path“ mit Türen und „2-buttons“

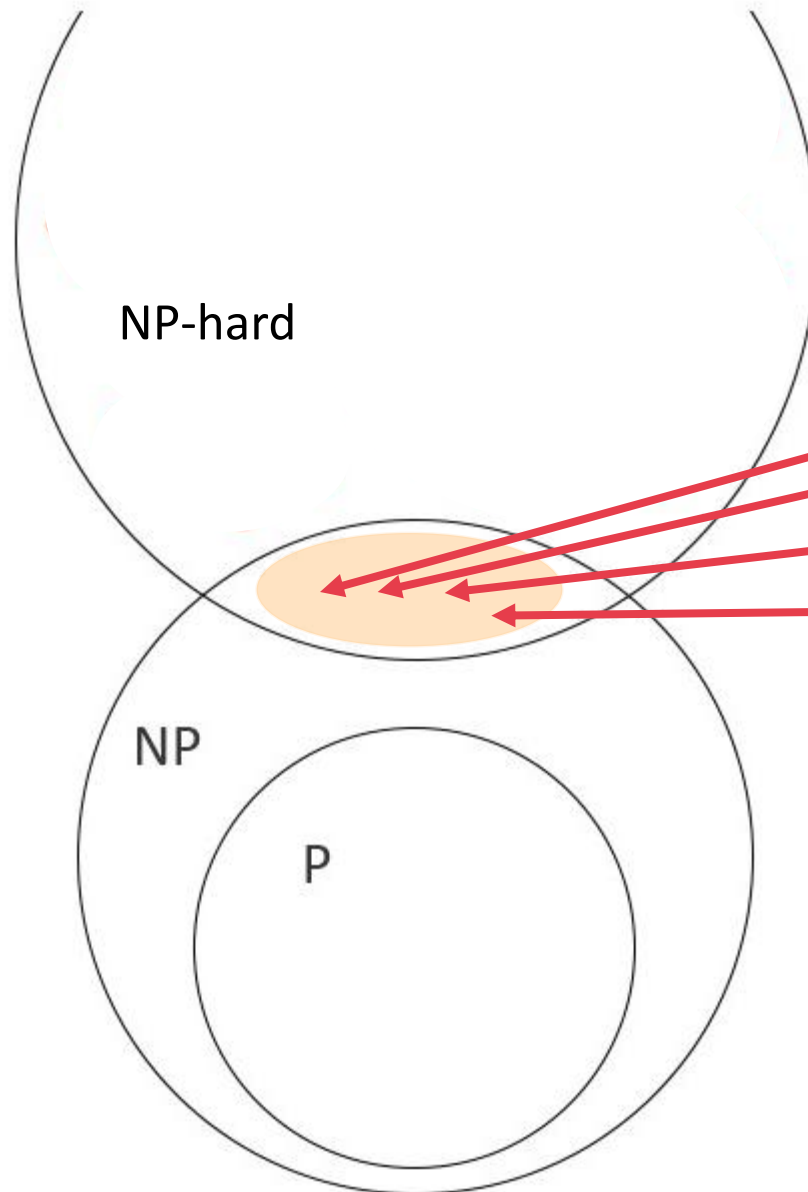


$SAT = \{F \mid F \text{ ist aussagenlogische Formel und erfüllbar}\}$

MINESWEEPER ist NP-vollständig

Hamiltonkreis ist NP-vollständig





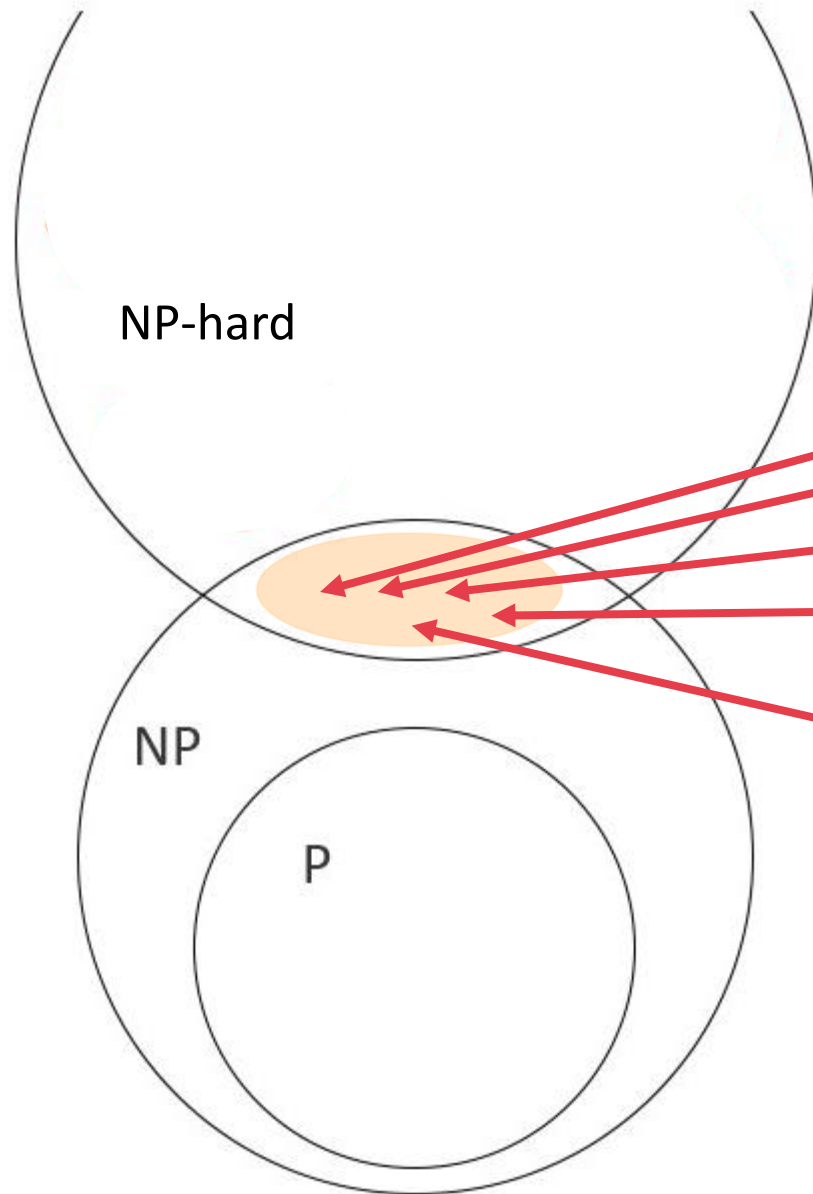
$$SAT = \{F \mid F \text{ ist aussagenlogische Formel und erfüllbar}\}$$

MINESWEEPER ist NP-vollständig

Hamiltonkreis ist NP-vollständig

Traveling Salesman ist NP-vollständig





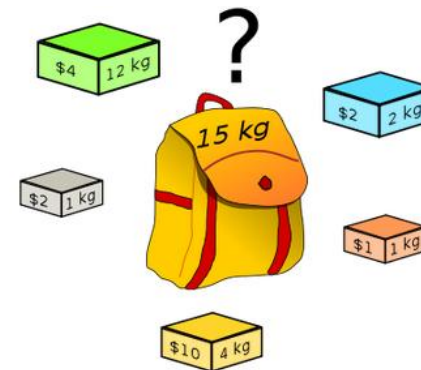
$SAT = \{F \mid F \text{ ist aussagenlogische Formel und erfüllbar}\}$

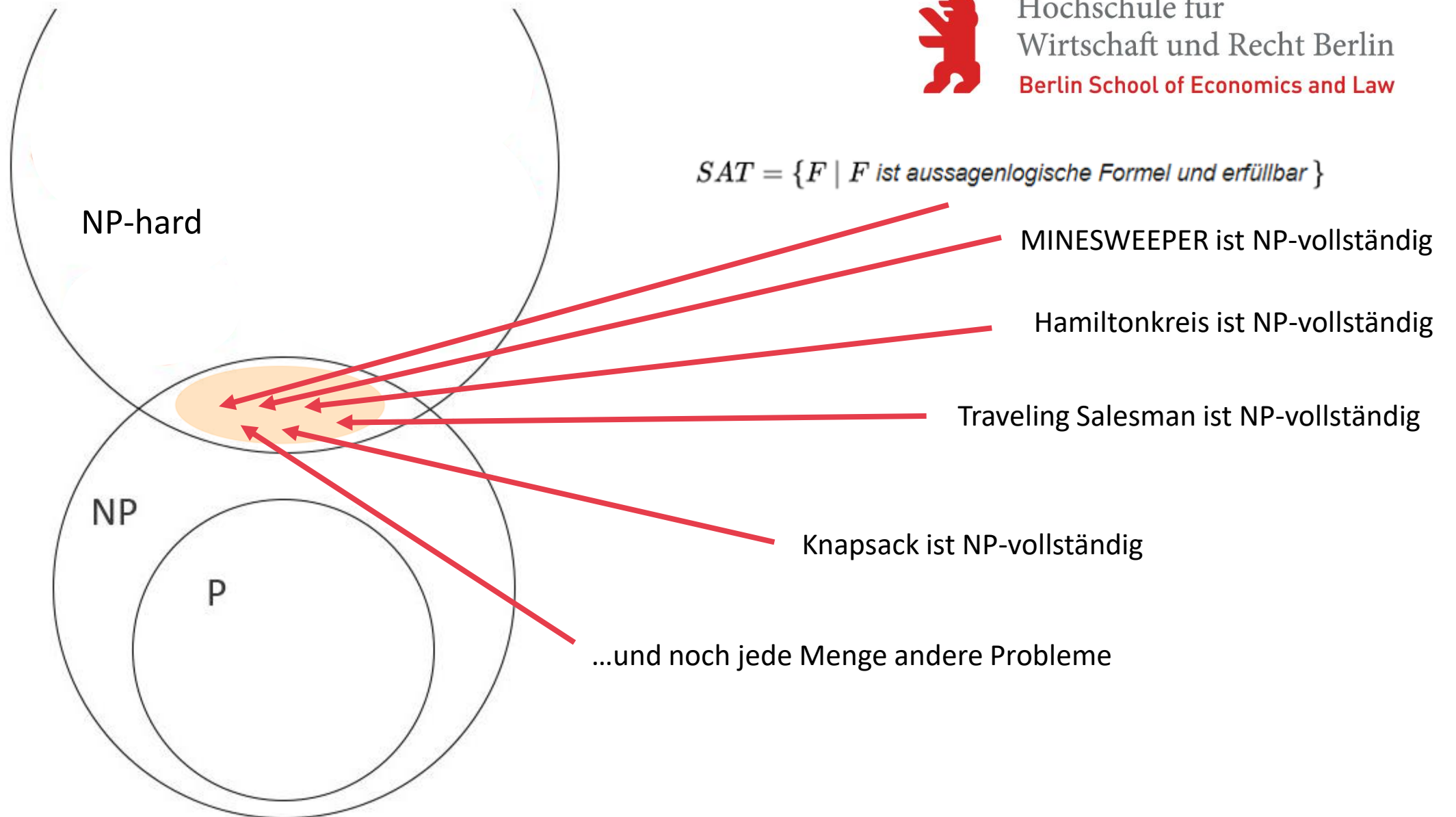
MINESWEEPER ist NP-vollständig

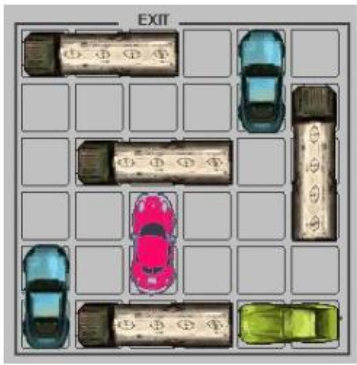
Hamiltonkreis ist NP-vollständig

Traveling Salesman ist NP-vollständig

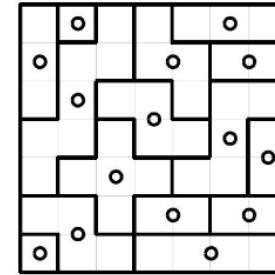
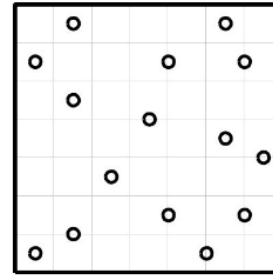
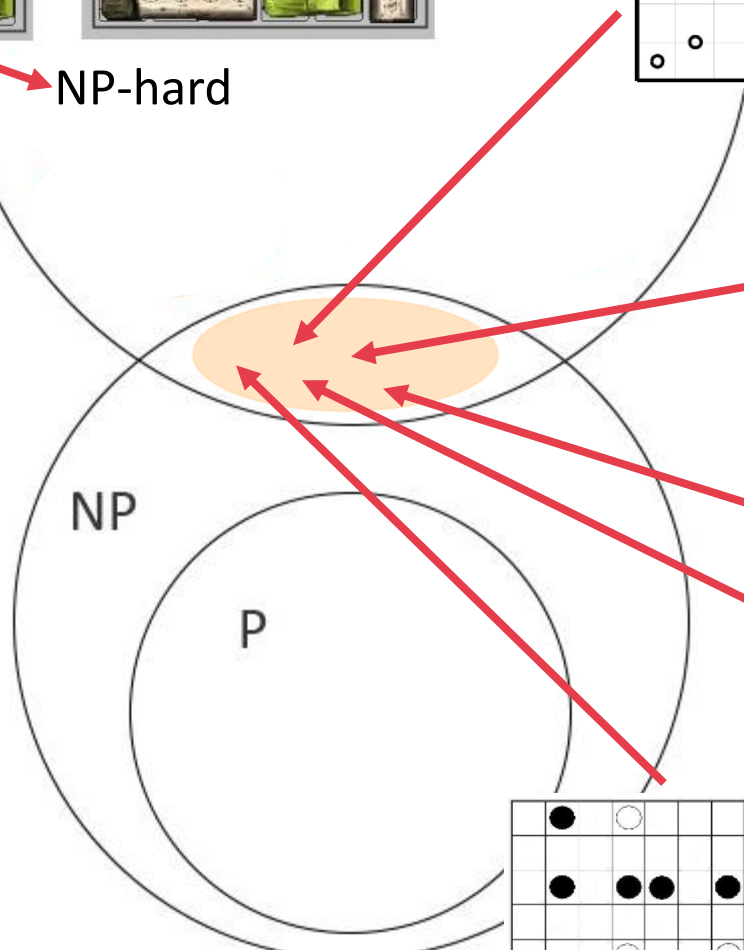
Knapsack ist NP-vollständig



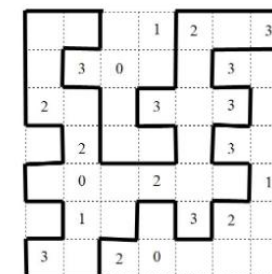
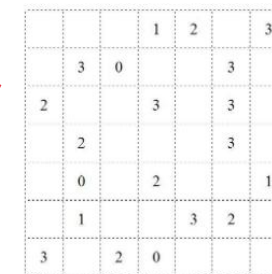
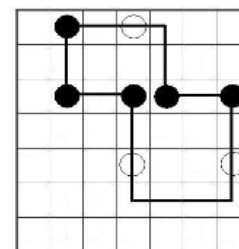
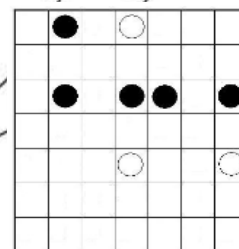
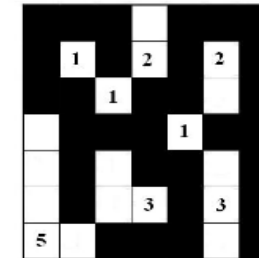
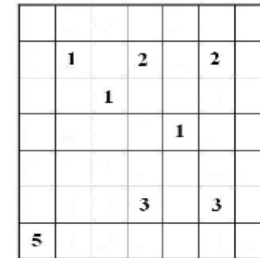
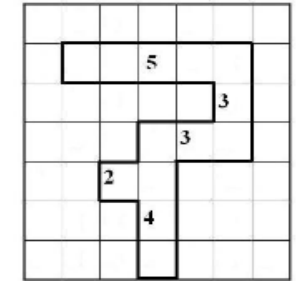
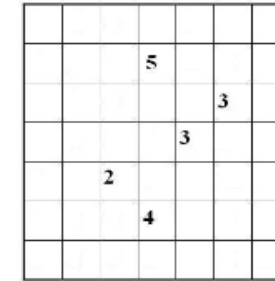




NP-hard



Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law



Karp's List

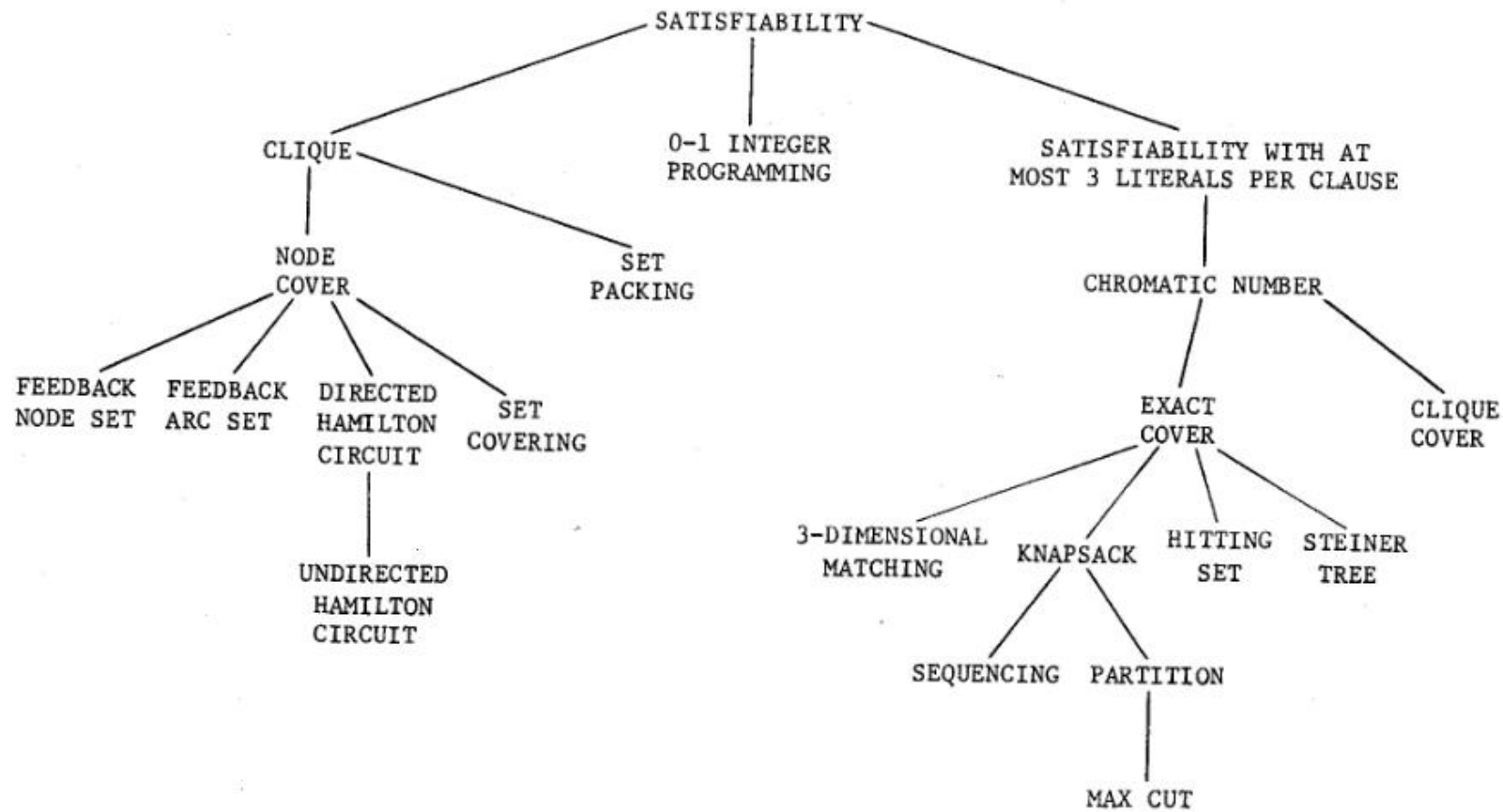


FIGURE 1 - Complete Problems



Space Complexity

Let M be a deterministic Turing machine that halts on all inputs. The **space complexity** of M is the function $f: \mathcal{N} \rightarrow \mathcal{N}$, where $f(n)$ is the **maximum number of tape cells** that M scans on any input of length n . If the space complexity of M is $f(n)$, we also say that M runs in space $f(n)$.

If M is a **nondeterministic** Turing machine wherein all branches halt on all inputs, we define its space complexity $f(n)$ to be the maximum number of tape cells that M scans on **any branch** of its computation for any input of length n .

(in der Literatur wird hier gelegentlich angenommen, dass die TM ein Eingabe- und ein Arbeitsband besitzt und dass **nur der die Zellen auf dem Arbeitsband** gezählt werden)



Space Complexity

Let $f: \mathcal{N} \rightarrow \mathcal{R}^+$ be a function. The **space complexity classes**, $\mathbf{SPACE}(f(n))$ and $\mathbf{NSPACE}(f(n))$, are defined as follows.

$\mathbf{SPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space deterministic Turing machine}\}.$

$\mathbf{NSPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space nondeterministic Turing machine}\}.$

Savitch's Theorem



THEOREM

Savitch's theorem For any function $f: \mathcal{N} \rightarrow \mathcal{R}^+$, where $f(n) \geq n$,
 $\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$.

*gilt auch für
 $f(n) \geq \log(n)$*





Savitch's Theorem

Für eine nichtdet. TM N , welche die Sprache A mit Speicher $f(n)$ entscheidet konstruieren wir eine detem. TM M , welche die Sprache A in $O(f(n)^2)$ entscheidet:

$M =$ "On input w :

1. Output the result of $\text{CANYIELD}(c_{\text{start}}, c_{\text{accept}}, 2^{df(n)})$."

Entscheide ob Konfig. c_2 von
Konfig. c_1 in t Schritten
erreicht werden kann

Da $f(n)$ zu Beginn nicht bekannt ist,
muss M nacheinander $f(n) = 1, 2, 3, \dots$
durchtesten

$\text{CANYIELD} =$ "On input c_1, c_2 , and t :

1. If $t = 1$, then test directly whether $c_1 = c_2$ or whether c_1 yields c_2 in one step according to the rules of N . *Accept* if either test succeeds; *reject* if both fail.
2. If $t > 1$, then for each configuration c_m of N on w using space $f(n)$:
3. Run $\text{CANYIELD}(c_1, c_m, \frac{t}{2})$.
4. Run $\text{CANYIELD}(c_m, c_2, \frac{t}{2})$.
5. If steps 3 and 4 both accept, then *accept*.
6. If haven't yet accepted, *reject*."

Jeder Level benötigt $O(f(n))$ Speicher

Die Rekursionstiefe beträgt $O(\log t)$,
und damit insgesamt $O(f(n)^2)$

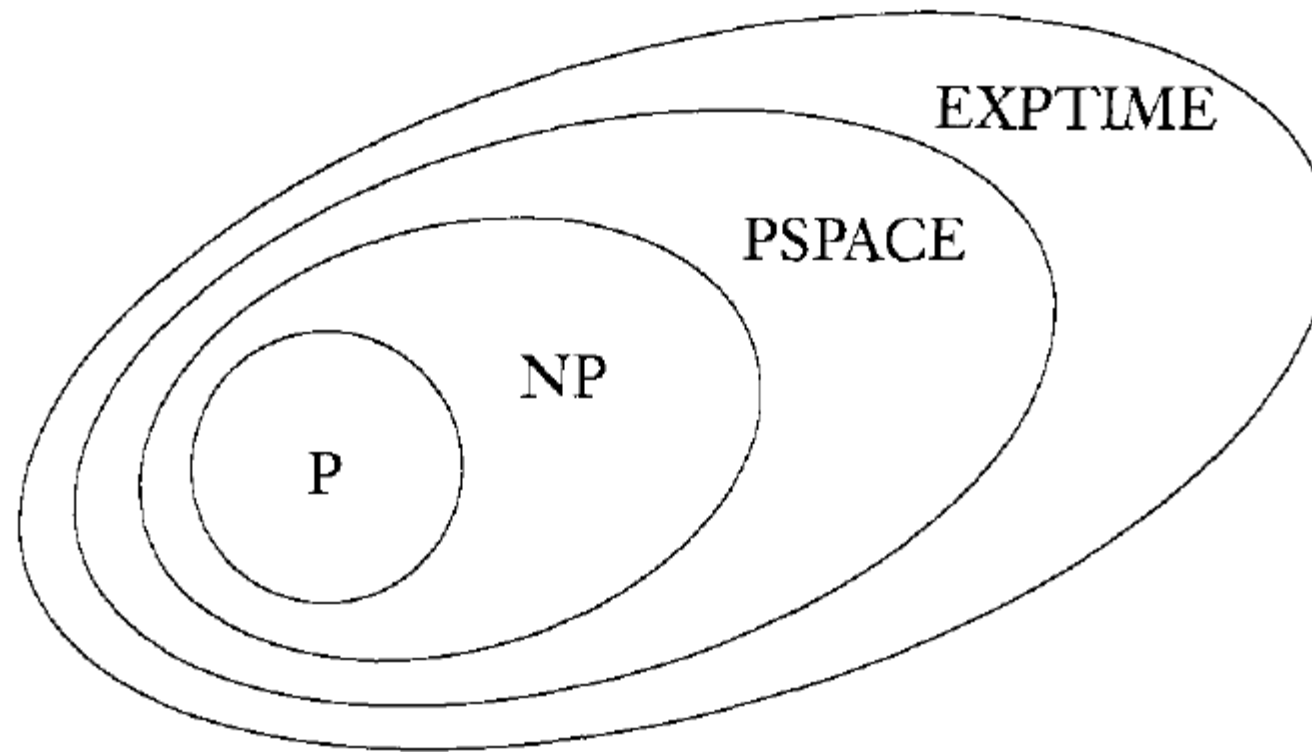
PSPACE



PSPACE is the class of languages that are decidable in **polynomial space on a deterministic** Turing machine. In other words,

$$\text{PSPACE} = \bigcup_k \text{SPACE}(n^k).$$

Time & Space



$$P \subseteq NP \subseteq PSPACE = NPSpace \subseteq EXPTIME = \bigcup_k \text{TIME}(2^{n^k})$$

PSPACE-Vollständigkeit



A language B is **PSPACE-complete** if it satisfies two conditions:

1. B is in PSPACE, and
2. every A in PSPACE is polynomial time reducible to B .

If B merely satisfies condition 2, we say that it is **PSPACE-hard**.

True Quantified Boolean Formulas



Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

$TQBF = \{\langle \phi \rangle \mid \phi \text{ is a true fully quantified Boolean formula}\}$

THEOREM

$TQBF$ is PSPACE-complete.