

# Logik

## Teil 4: Prädikatenlogik zweiter Stufe



# Übersicht Teil 4

Next



## 4.1 Syntax und Semantik

4.2 Eigenschaften

4.3 Nachbemerkungen



# Logik zweiter Stufe

Die Logik zweiter Stufe (SO) erweitert die Logik erster Stufe, behebt die meisten Unzulänglichkeiten in der Ausdrucksstärke.

## Grundidee:

Man kann nicht nur über **Elemente** von Strukturen quantifizieren, sondern auch über **Relationen beliebiger Stelligkeit**.

Beachte: **einstellige Relationen = Mengen von Elementen**

Definierbar werden dadurch beispielsweise:

- Azyklizität
- Graphen, die ein Baum sind
- Planarität
- $k$ -Färbbarkeit für beliebiges (fixes)  $k \geq 2$
- etc



# Logik zweiter Stufe

## Die Logik zweiter Stufe

- hat eine sehr **befriedigende Ausdrucksstärke**
- hängt eng mit anderen Gebieten der Informatik zusammen, insbes. den formalen Sprachen und der Komplexitätstheorie

Den Vorteilen stehen (noch) schlechtere Berechnungseigenschaften als in FO gegenüber.

SO sollte als Logik zur **Definition interessanter Eigenschaften** betrachtet werden, weniger als Logik zum automatischen Schließen.

Wir fixieren für jedes  $n \geq 1$  eine abzählbar unendliche Menge  $\text{VAR}^n$  von ***n*-ären Relationsvariablen**.

Diese bezeichnen wir meist mit  $X$ .

T4.1



# Syntax

## Definition 4.1 (SO-Formeln)

Die Menge der **Formeln** der Prädikatenlogik **zweiter Stufe** ist induktiv wie folgt definiert:

- Sind  $t_1, t_2$  Terme, dann ist  $t_1 = t_2$  eine Formel.
- Sind  $t_1, \dots, t_n$  Terme und  $P$  ein  $n$ -stelliges Relationssymbol, dann ist  $P(t_1, \dots, t_n)$  eine Formel.
- Sind  $t_1, \dots, t_n$  Terme und  $X \in \text{VAR}^n$ , dann ist  $X(t_1, \dots, t_n)$  eine Formel.
- Wenn  $\varphi$  und  $\psi$  Formeln sind, dann auch  $\neg\varphi$ ,  $(\varphi \wedge \psi)$ ,  $(\varphi \vee \psi)$ .
- Wenn  $\varphi$  eine Formel ist und  $x \in \text{VAR}$ , dann sind  $\exists x \varphi$  und  $\forall x \varphi$  Formeln.
- Wenn  $\varphi$  eine Formel ist und  $X \in \text{VAR}^n$ , dann sind  $\exists X \varphi$  und  $\forall X \varphi$  Formeln.

Die Menge aller SO-Formeln über einer Signatur  $\tau$  bezeichnen wir mit  $\text{SO}(\tau)$ .



# Sprechweisen und Konventionen

- Atome haben nun drei mögliche Formen:

$$t = t' \quad P(t_1, \dots, t_n) \quad X(t_1, \dots, t_n)$$

- Die Quantoren  $\exists X$  und  $\forall X$  binden genau wie  $\exists x$  und  $\forall x$  (stärker als  $\wedge$  und  $\vee$ , die wiederum stärker als  $\rightarrow$ ,  $\leftrightarrow$ ).
- Relationsvariablen können ebenso wie Objektvariablen frei oder gebunden vorkommen.
- Ein Satz ist eine Formel ohne freie Variablen beider Arten.



# Semantik

Wir gehen wieder in zwei Schritten vor: zunächst Terme, dann Formeln

## Definition 4.2 (SO-Zuweisung, SO-Semantik)

Sei  $\mathfrak{A}$  eine Struktur. Eine **SO-Zuweisung** in  $\mathfrak{A}$  ist eine Abbildung  $\beta$ , die

- jeder Objektvariablen  $x \in \text{VAR}$  ein Element  $\beta(x) \in A$  und
- jeder  $n$ -ären Relationsvariablen  $X \in \text{VAR}^n$  eine  
 $n$ -äre Relation  $\beta(X) \subseteq A^n$

zuweist. Wie in FO erweitert man  $\beta$  induktiv auf Terme.

Erweiterung der Erfülltheitsrelation  $\models$  auf SO:

- $\mathfrak{A}, \beta \models X(t_1, \dots, t_n)$  gdw.  $(\beta(t_1), \dots, \beta(t_n)) \in \beta(X)$
- $\mathfrak{A}, \beta \models \exists X \varphi$  mit  $X \in \text{VAR}^n$  gdw. ein  $R \subseteq A^n$  existiert mit  $\mathfrak{A}, \beta[X/R] \models \varphi$
- $\mathfrak{A}, \beta \models \forall X \varphi$  mit  $X \in \text{VAR}^n$  gdw. für alle  $R \subseteq A^n$  gilt, dass  $\mathfrak{A}, \beta[X/R] \models \varphi$



# Relationssymbole vs Relationsvariablen

Beachte Unterschied zwischen Relationssymbolen und Relationsvariablen:

- Relations**symbole** gehören zur Signatur und werden durch **Struktur** interpretiert

Z.B.:  $\exists x \exists y R(x, y)$  drückt aus:  
„Es gibt eine *R*-Kante (in der Struktur)“

- Relations**variablen** gehören **nicht** zur Signatur und sind in der Struktur **nicht** „sichtbar“

Z.B.:  $\exists x \exists y \exists X X(x, y)$  ist Tautologie!  
 $\exists x \exists y \forall X X(x, y)$  ist ???



# Relationssymbole vs Relationsvariablen

Beachte Unterschied zwischen Relationssymbolen und Relationsvariablen:

- Relations**symbole** gehören zur Signatur und werden durch **Struktur** interpretiert

Z.B.:  $\exists x \exists y R(x, y)$  drückt aus:  
„Es gibt eine *R*-Kante (in der Struktur)“

- Relations**variablen** gehören **nicht** zur Signatur und sind in der Struktur **nicht** „sichtbar“

Z.B.:  $\exists x \exists y \exists X X(x, y)$  ist Tautologie!

$\exists x \exists y \forall X X(x, y)$  ist unerfüllbar!



# Beispiele

Seit  $\tau = \{E\}$  die Signatur gerichteter Graphen

## Erreichbarkeit:

$$\varphi_{\text{reach}}(x, y) = \forall X \left( \left( \underbrace{X(x)}_{\rightarrow} \wedge \forall z \forall z' (X(z) \wedge E(z, z') \rightarrow X(z')) \right) \rightarrow X(y) \right)$$

1-stellig

„Jede Knotenmenge, die  $x$  enthält und unter Nachfolgern abgeschlossen ist, enthält auch  $y$ .“

T4.2

Für all  $\tau$ -Strukturen  $\mathfrak{A}$  und Elemente  $a, b \in A$  gilt:

$\mathfrak{A} \models \varphi_{\text{reach}}(a, b)$  gdw.  $b$  in gerichtetem Graph  $\mathfrak{A}$   
von  $a$  aus erreichbar ist

Zusammenhang gerichteter und ungerichteter Graphen dann natürlich auch ausdrückbar



# Beispiele

**EVEN** = {  $\mathfrak{A}$  |  $|A|$  geradzahlig oder unendlich }

$$\text{Func}(R) = \forall x \forall y \forall y' ( R(x, y) \wedge R(x, y') \rightarrow y = y' )$$

$$\text{Func}^-(R) = \forall x \forall y \forall y' ( R(y, x) \wedge R(y', x) \rightarrow y = y' )$$

$$\varphi_{\text{even}} = \exists R ( \forall x \exists y R(x, y) \vee R(y, x) \\ \wedge \forall x ( \exists y R(x, y) \rightarrow \neg \exists y R(y, x) ) \\ \wedge \text{Func}(R) \wedge \text{Func}^-(R) )$$

2-stellig

„ $A$  kann ohne Überlappungen mit  $R \in \text{VAR}^2$  überdeckt werden.“

**T4.3**

Für alle Strukturen  $\mathfrak{A}$  gilt:

$$\mathfrak{A} \models \varphi_{\text{even}} \text{ gdw. } \mathfrak{A} \in \text{EVEN}$$



# Beispiele

## Unendliche Strukturen:

$$\varphi_\infty = \exists R \left( \begin{array}{l} \exists x \exists y R(x, y) \wedge \\ \forall x \forall y (R(x, y) \rightarrow \exists z R(y, z)) \wedge \\ \forall x \neg R(x, x) \wedge \\ \forall x \forall y \forall z ((R(x, y) \wedge R(y, z)) \rightarrow R(x, z)) \end{array} \right)$$

2-stellig

„Man kann eine Teilmenge des Universums in einer irreflexiven, transitiven Ordnung ohne größtes Element anordnen.“

T4.4

Für alle Strukturen  $\mathfrak{A}$  gilt:  $\mathfrak{A} \models \varphi$  gdw.  $|A|$  unendlich

Endliche Strukturen:  $\neg \varphi_\infty$

Auch Abzählbarkeit und Überabzählbarkeit sind definierbar.

Löwenheim-Skolem gilt also **nicht** (weder aufwärts noch abwärts).



# Beispiele

## Kompaktheit gilt ebenfalls nicht:

Betrachte folgende Menge von SO-Sätzen.

$$\Gamma = \{\neg\varphi_\infty\} \cup \{\varphi_n \mid n \geq 1\}$$

wobei  $\varphi_n = \exists x_1 \cdots \exists x_n \bigwedge_{1 \leq i < j \leq n} x_i \neq x_j$  „Die Struktur hat Größe  $\geq n$ .“

Offensichtlich:

- $\Gamma$  ist unerfüllbar
- Jede endliche Teilmenge  $\Delta \subseteq \Gamma$  ist erfüllbar  
(in einer Struktur der Größe  $\max\{n \mid \varphi_n \in \Delta\}$ )

SO hat also **nicht** die Kompaktheits-Eigenschaft.



# Beispiele

Betrachte Signatur mit Konstantensymbol 0, unärem Funktionssymbol nf

**Die Peano-Axiome** (in leicht angepasster Form):

$$\alpha_1: \forall x \text{nf}(x) \neq 0$$

$$\alpha_2: \forall x \forall y (\text{nf}(x) = \text{nf}(y) \rightarrow x = y)$$

$$\alpha_3: \forall X \left( (X(0) \wedge \forall x (X(x) \rightarrow X(\text{nf}(x)))) \rightarrow \forall x X(x) \right)$$

## Lemma 4.3

$\mathfrak{A} \models \{\alpha_1, \alpha_2, \alpha_3\}$  **gdw.**  $\mathfrak{A}$  isomorph zu  $(\mathbb{N}, \text{nf}, 0)$ .

Wegen des aufsteigenden Satzes von Löwenheim-Skolem für FO  
gibt es keine FO-Formel, die das leistet!

In SO lassen sich basierend auf 0 und nf auch 1, + und  $\cdot$  definieren;  
also lässt sich Arithmetik bis auf Isomorphie definieren!



**Folgende Resultate überträgt man leicht von FO nach SO:**

- Koinzidenzlemma und Isomorphielemma
- Existenz äquivalenter Formeln in Pränex-Normalform  
(alle herstellbar in Linearzeit)

**Die Dualität der Quantoren gilt auch für SO-Quantoren:**

$$\neg \exists R \neg \varphi \equiv \forall R \varphi$$

für Relationsvariablen  $R$

$$\neg \forall R \neg \varphi \equiv \exists R \varphi$$

beliebiger Stelligkeit



# MSO

Für viele Zwecke genügen bereits **unäre Relationsvariablen**.

Das resultierende MSO hat bessere Eigenschaften als das volle SO.

## Definition 4.4 (Monadische Logik zweiter Stufe, MSO)

Eine SO-Formel  $\varphi$  heißt **monadisch**,  
wenn sie keine Relationsvariablen mit Stelligkeit  $> 1$  enthält.

**MSO** ist die Menge aller monadischen SO-Formeln.

**Beispiel:** Zusammenhang ungerichteter Graphen ist MSO-ausdrückbar.

$$\begin{aligned}\varphi_{\text{conn}} = \forall X \Big( & \Big( \exists x X(x) \wedge \\ & \forall x \forall y \big( (X(x) \wedge E(x, y)) \rightarrow X(y) \big) \wedge \\ & \forall x \forall y \big( (X(x) \wedge E(y, x)) \rightarrow X(y) \big) \Big) \\ & \rightarrow \forall x X(x) \Big)\end{aligned}$$

„Jede Zusammenhangskomponente enthält alle Knoten.“



Next

4.1 Syntax und Semantik

## 4.2 Eigenschaften

4.3 Nachbemerkungen



# Auswertungsproblem

Der Algorithmus für das Auswerten von FO-Formeln kann leicht auf SO-Formeln erweitert werden.

Eine Komplexitätsanalyse zeigt aber wichtige Unterschiede!



# Zur Erinnerung: der Algorithmus für FO

**ausw( $\mathfrak{A}, \beta, \varphi$ )**

**case**

$\varphi = (t = t')$ : **return 1 if  $\beta(t) = \beta(t')$ , else return 0**

$\varphi = P(t_1, \dots, t_k)$ : **return 1 if  $(\beta(t_1), \dots, \beta(t_k)) \in P^{\mathfrak{A}}$ , else return 0**

$\varphi = \neg\psi$ : **return 1 - ausw( $\mathfrak{A}, \beta, \psi$ )**

$\varphi = \psi \wedge \vartheta$ : **return min{ausw( $\mathfrak{A}, \beta, \psi$ ), ausw( $\mathfrak{A}, \beta, \vartheta$ )}**

$\varphi = \psi \vee \vartheta$ : **return max{ausw( $\mathfrak{A}, \beta, \psi$ ), ausw( $\mathfrak{A}, \beta, \vartheta$ )}**

$\varphi = \exists x \psi$ :

**rufe ausw( $\mathfrak{A}, \beta[x/a], \psi$ ) für alle  $a \in A$**

**return 1 if ein Ruf erfolgreich, else return 0**

$\varphi = \forall x \psi$ :

**rufe ausw( $\mathfrak{A}, \beta[x/a], \psi$ ) für alle  $a \in A$**

**return 1 if alle Rufe erfolgreich, else return 0**

**endcase**



# Erweiterung des Algorithmus auf SO

## Fälle für die SO-Teilformeln:

case

$\varphi = X(t_1, \dots, t_\ell)$ : return 1 if  $(\beta(t_1), \dots, \beta(t_\ell)) \in \beta(X)$ ,  
else return 0

$\varphi = \exists X \psi$  wobei  $X$  eine  $\ell$ -stellige Relationsvariable:  
rufe ausw( $\mathfrak{A}, \beta[X/B], \psi$ ) für alle  $B \subseteq A^\ell$   
return 1 if ein Ruf erfolgreich, else return 0

$\varphi = \forall X \psi$  wobei  $X$  eine  $\ell$ -stellige Relationsvariable:  
rufe ausw( $\mathfrak{A}, \beta[X/B], \psi$ ) für alle  $B \subseteq A^\ell$   
return 1 if alle Rufe erfolgreich, else return 0

endcase



# Zeitkomplexität von Auswertung

## Lemma 4.6

Bei Eingabe einer Struktur  $\mathfrak{A}$  der Größe  $n$  und einer Formel  $\varphi$  der Größe  $k$  benötigt der Algorithmus

1. Zeit  $\mathcal{O}(n^k)$ , wenn  $\varphi$  eine FO-Formel ist
2. Zeit  $2^{\mathcal{O}(nk)}$ , wenn  $\varphi$  eine MSO-Formel ist
3. Zeit  $2^{\mathcal{O}(n^{2k})}$  im Allgemeinen.

T4.5

Diese Komplexitäten kann man (wahrscheinlich) nicht wesentlich verbessern.

**Beachte:** in Anwendungen ist ...

- $n$  meist sehr groß (Datenbank, zu verifizierendes System, ...)
- $k$  meist eher klein (Datenbankanfrage, zu verifizierende Eigenschaft, ...)

Diese Beobachtungen führen zur Betrachtung der Datenkomplexität.



# Datenkomplexität von Auswertung

Bei der Datenkomplexität betrachtet man

- nur die Struktur  $\mathfrak{A}$  als Eingabe (die Datenbank bzw. das System)
- die Formel  $\varphi$  als fixiert, also ist ihre Größe  $k$  eine Konstante.

Der Algorithmus liefert dann folgende Datenkomplexität:

## Lemma 4.7

Bei Eingabe einer Struktur  $\mathfrak{A}$  der Größe  $n$  benötigt der Algorithmus

1. Zeit  $\text{poly}(n)$  wenn eine FO-Formel ausgewertet wird
2. Zeit  $2^{\mathcal{O}(n)}$  wenn eine MSO-Formel ausgewertet wird
3. Zeit  $2^{\text{poly}(n)}$  wenn eine SO-Formel ausgewertet wird

Aus dieser Perspektive

verhält sich FO also deutlich besser als (M)SO - darum basiert SQL auf FO!



# Komplexität von Gültigkeit

**Gültigkeit** ist in SO ein **noch schwierigeres** Problem als in FO:

## Theorem 4.8

Die Menge der SO Tautologien ist nicht rekursiv aufzählbar.

Auch die **erfüllbaren** Formeln und **unerfüllbaren** Formeln sind natürlich nicht rekursiv aufzählbar; all das gilt **bereits in MSO**.

Das bedeutet auch:

Es gibt **keine vollständigen Kalküle** für (M)SO

Damit auch **automatisches Theorembeweisen** im Sinne von FO unmöglich.  
Es gibt dennoch SO-Beweiser (wie **Isabelle** oder **HOL**),  
die aber vom Benutzer „geleitet“ werden müssen.



# Komplexität von Gültigkeit

## Theorem 4.8

Die Menge der SO Tautologien ist nicht rekursiv aufzählbar.

### Beweis:

Wir zeigen: Rekursive Aufzählbarkeit SO-Tautologien  
⇒ rekursive Aufzählbarkeit erfüllbare FO-Sätze

Dabei betrachten wir der Einfachheit halber FO-Sätze über  $\tau = \{E\}$

Erfüllbare FO-Sätze auch in dieser Signatur nicht rekursiv aufzählbar

Unser Argument generalisiert aber auch leicht auf beliebige Signaturen.



# Komplexität von Gültigkeit

## Theorem 4.8

Die Menge der SO Tautologien ist nicht rekursiv aufzählbar.

### Beweis:

Sei  $\varphi \in \text{FO}(\tau)$  mit  $\tau = \{E\}$ . Es gilt:

1.  $\varphi$  hat Modell der Größe  $n \in \mathbb{N}$  gdw.  $\varphi_n \rightarrow \exists E \varphi$  gültig

wobei  $\varphi_n = \exists x_1 \cdots \exists x_n \left( \bigwedge_{1 \leq i < j \leq n} x_i \neq x_j \wedge \forall y \bigvee_{1 \leq i \leq n} y = x_i \right)$

T4.6

2.  $\varphi$  hat unendliches Modell gdw.  $\varphi_\infty \rightarrow \exists E \varphi$  gültig

↑  
siehe SO-Beispiele



# Komplexität von Gültigkeit

## Theorem 4.8

Die Menge der SO Tautologien ist nicht rekursiv aufzählbar.

### Beweis:

Sei  $\varphi \in \text{FO}(\tau)$  mit  $\tau = \{E\}$ . Es gilt:

1.  $\varphi$  hat Modell der Größe  $n \in \mathbb{N}$  gdw.  $\varphi_n \rightarrow \exists E \varphi$  gültig

wobei  $\varphi_n = \exists x_1 \cdots \exists x_n \left( \bigwedge_{1 \leq i < j \leq n} x_i \neq x_j \wedge \forall y \bigvee_{1 \leq i \leq n} y = x_i \right)$

T4.6

2.  $\varphi$  hat unendliches Modell gdw.  $\varphi_\infty \rightarrow \exists E \varphi$  gültig

Man kann also wie folgt die erfüllbaren FO-Sätze aufzählen:

- Zähle alle gültigen SO-Formeln auf
- Für jede erzeugte SO-Formel der Form  $\varphi_n \rightarrow \exists E \varphi$  und  $\varphi_\infty \rightarrow \exists E \varphi$  mit  $\varphi$   $\text{FO}(\tau)$ -Formel: gib  $\varphi$  aus

□



4.1 Syntax und Semantik

4.2 Eigenschaften

Next  **4.3 Nachbemerkungen**



# SO in der Informatik (1)

In VL **Automaten und Sprachen** werdet Ihr wichtige Klassen von **formalen Sprachen** kennenlernen.

Eine besonders wichtige und natürliche sind die sog. **regulären Sprachen**

## Theorem 4.9 (Büchi-Elgot-Trakhtenbrot)

Für jede formale Sprache  $L$  sind äquivalent:

1.  $L$  ist regulär
2.  $L$  ist **definierbar** durch einen MSO-Satz

Also: enger Zusammenhang Logik  $\Leftrightarrow$  Formale Sprachen

(wird in AFS üblicherweise nicht behandelt, aber in MSc-Vorlesungen)



# SO in der Informatik (2)

In (M)SO lassen sich viele wichtige algorithmische Probleme definieren

Der Algorithmus für das Auswertungsproblem löst dann alle diese Probleme

Es ergibt sich auch ein enger Zusammenhang zur Komplexitätstheorie

Eine wichtige und natürliche Komplexitätsklasse ist NP

(Die Hauptzutat zur berühmten P = NP Frage)

## Theorem 4.10 (Fagin)

Für jedes Problem  $L$  sind äquivalent:

1.  $P$  ist in NP
2.  $P$  ist definierbar durch einen existentiellen SO-Satz

(Wird in Berechenbarkeit nicht behandelt, aber in MSc-Vorlesungen)



# Beispiel: 3-Färbbarekeit

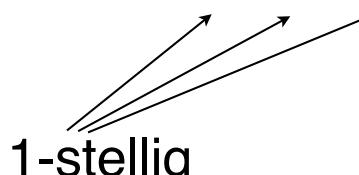
Beispiele für die Fähigkeit von (M)SO, algorithmische Probleme zu definieren

## Definition 4.11 (3-Färbbarekeit)

Ein ungerichteter Graph  $G = (V, E)$  heißt **3-färbbare**, wenn es eine Abbildung  $f : V \rightarrow \{r, g, b\}$  gibt, so dass gilt:

- Für alle  $\{v_1, v_2\} \in E$  ist  $f(v_1) \neq f(v_2)$ .

**Betrachte**  $\varphi_{3F} = \exists C_1 \exists C_2 \exists C_3 \left( \forall x \forall y \left( (C_1(x) \vee C_2(x) \vee C_3(x)) \wedge \bigwedge_{1 \leq i < j \leq 3} \neg(C_i(x) \wedge C_j(x)) \wedge E(x, y) \rightarrow \bigwedge_{1 \leq i \leq 3} \neg(C_i(x) \wedge C_i(y)) \right) \right)$



1-stellig

Es gilt:

$$G \models \varphi_{3F} \text{ gdw. } G \text{ 3-färbbare} \quad \text{für alle ungerichteten Graphen } G$$

# Beispiel: Hamiltonkreis

## Definition 4.12 (Hamiltonkreis)

Sei  $G = (V, E)$  ein ungerichteter Graph. Ein **Hamiltonkreis** in  $G$  ist ein geschlossener Pfad, der jeden Knoten genau einmal enthält.

$$\varphi_{HK} = \exists R (\forall x \forall y (R(x, y) \rightarrow E(x, y)) \wedge$$

(mit  $R$  anstelle von  $E$ )

2-stellig  $\varphi_{conn} \wedge$

$$\forall x \exists y R(x, y) \wedge$$
$$\forall x \exists y \exists z (R(x, y) \wedge R(x, z) \wedge y \neq z \wedge$$
$$\forall u (R(x, u) \rightarrow (x = y \vee x = z)))$$

„Es gibt eine Teilmenge  $R$  der Kanten, die alle Knoten enthält, zusammenhängend ist und in der Es gilt: jeder Knoten genau zwei Nachbarn hat.“

T4.7

$G \models \varphi_{HK}$  gdw.  $G$  enthält Hamiltonkreis für alle ungerichteten Graphen  $G$

# Zurück zum Auswertungsproblem

Also:

Das Auswertungsproblem für MSO löst auch 3-Färbbarkeit

SO löst auch Hamiltonkreis

Dabei ist die Formel **fest**, ihre Größe also eine Konstante

⇒ Datenkomplexität

3F also in Zeit  $2^{\mathcal{O}(n)}$  lösbar, HK in Zeit  $2^{\text{poly}(n)}$

Das kann man auch sehr einfach direkt zeigen.

Die **uniforme Sichtweise** durch (M)SO auf viele verschiedene Probleme  
gibt aber Anlass für zahlreiche interessante Fragen

# Fertig für dieses Semester! :)

Dank u allen!

Thank you!

Kiitos!

Vielen Dank für  
Eure Aufmerksamkeit!

Спасибі!

¡Gracias!

Merci !

Dziękuję!

Спасибо!

“Contrariwise”, continued Tweedledee, “if it was so, it might be; and if it were so, it would be; but as it isn’t, it ain’t. That’s logic.”

aus Alice in Wonderland

