Carsten Knoll, Julius Fiedler
TU Dresden, Institut für Regelungs- und Steuerungstheorie

ERK & OCSE: Ein experimenteller Ansatz zur formalen Wissensrepräsentation der Regelungstheorie

GMA-FA 1.40, Anif, 19. September 2022

Motivation und Rückblick (1)

Regelungstheorie hat heterogenes Methodenspektrum

- PT1-Glied, ..., nichtlineare PDEs, ...

Motivation und Rückblick (1)

Regelungstheorie hat heterogenes Methodenspektrum

- PT1-Glied, ..., nichtlineare PDEs, ... $\Rightarrow$ z. T. hochspezialisierte Mathematik

Motivation und Rückblick (1)

Regelungstheorie hat heterogenes Methodenspektrum

- PT1-Glied, ..., nichtlineare PDEs, ... $\Rightarrow$ z. T. hochspezialisierte Mathematik

Regelungstheorie hat heterogenes Anwendungsspektrum

- Verfahrens-, Fahrzeug-, Gebäudetechnik, Robotik, ...

Beständiger Wissenszuwachs

- überlineares Wachstum der Publikationen $\rightarrow$ Spezialisierung notwendig

Motivation und Rückblick (1)

Regelungstheorie hat heterogenes Methodenspektrum

- PT1-Glied, ..., nichtlineare PDEs, ... $\Rightarrow$ z. T. hochspezialisierte Mathematik

Regelungstheorie hat heterogenes Anwendungsspektrum

- Verfahrens-, Fahrzeug-, Gebäudetechnik, Robotik, ...

Beständiger Wissenszuwachs

- überlineares Wachstum der Publikationen $\rightarrow$ Spezialisierung notwendig

$\Rightarrow$ **Wissenstransfer: nichttrivial**

- ... innerhalb der Regelungstechnik
- ... in Anwendungsdomänen

Thesen

These 1

Die gegenwärtig dominierenden Medien (Fließtext, Formeln, Grafiken) der Wissensrepräsentation sind in Bezug auf die angesprochenen Probleme suboptimal.

Thesen

These 1

Die gegenwärtig dominierenden Medien (Fließtext, Formeln, Grafiken) der Wissensrepräsentation sind in Bezug auf die angesprochenen Probleme suboptimal.

These 2

Formale Wissensrepräsentationsmethoden stellen eine vielversprechende Ergänzung dar.

Gliederung

☑ Motivation und Rückblick
→ **Formale Wissensrepräsentation: Ist-Stand**
☐ Ansatz: Emergent Representation of Knowledge (ERK)
☐ Ontology of Control Systems Engineering (OCSE)
☐ Zusammenfassung und Diskussion

Formale Wissensrepräsentation (1): Begriffe

Eine Ontologie [nach *Studer et. al. 1998*]

Maschinenverarbeitbare Spezifikation der begrifflichen Abdeckung einer Wissensdomäne.

→ Welche Begriffe gibt es? In welcher Beziehung stehen sie zueinander?

Formale Wissensrepräsentation (1): Begriffe

Eine Ontologie [nach _Studer et. al. 1998_]

Maschinenverarbeitbare Spezifikation der begrifflichen Abdeckung einer Wissensdomäne.

→ Welche Begriffe gibt es? In welcher Beziehung stehen sie zueinander?

Taxonomie

Hierarchisches Klassifikationssystem von `ist-ein`-Beziehungen

Beispiel: `Hauskatze` → `Säugetier` → `Wirbeltier` → `Lebewesen`

Formale Wissensrepräsentation (2): Wissensgraphen und RDF

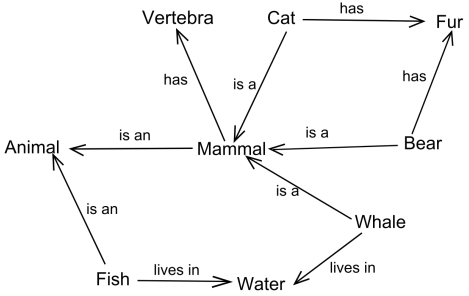
„Knowledge Graph“:

- Knoten: Begriffe
- Kanten: Beziehungen

Formale Wissensrepräsentation (2): Wissensgraphen und RDF

„Knowledge Graph“:

- Knoten: Begriffe
- Kanten: Beziehungen

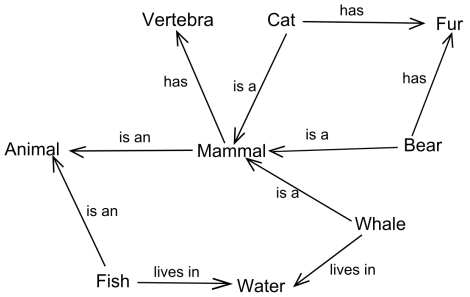


Quelle: wikipedia.org/... (CC0)

Formale Wissensrepräsentation (2): Wissensgraphen und RDF

„Knowledge Graph“:

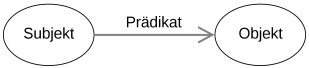
- Knoten: Begriffe
- Kanten: Beziehungen



Quelle: wikipedia.org/... (CC0)

Ressource **D**escription **F**ramework:

- Sprache zur Beschreibung von Subjekt-Prädikat-Objekt-Tripeln



- Zugehörige Abfragesprache:
  SPARQL
  (SPARQL Protocol And RDF Query Language)

Formale Wissensrepräsentation (3): OWL und Inferenz

Web **O**ntology **L**anguage

- OWL2: definierter Standard; basiert auf RDF
- Theoretische Basis: Beschreibungslogik(en)
  - „Profile“ mit Unterschiedlicher Ausdrucksstärke
  - Entscheidbare Fragmente der Prädikatenlogik 1. Stufe

  $\Rightarrow$ Einfluss auf Komplexität von Inferenz-Algorithmen

Formale Wissensrepräsentation (3): OWL und Inferenz

Web **O**ntology **L**anguage

- OWL2: definierter Standard; basiert auf RDF
- Theoretische Basis: Beschreibungslogik(en)
  - „Profile“ mit Unterschiedlicher Ausdrucksstärke
  - Entscheidbare Fragmente der Prädikatenlogik 1. Stufe

  $\Rightarrow$ Einfluss auf Komplexität von Inferenz-Algorithmen

Inferenzsystem („Schließer“ bzw. Reasoner)

- Kann Schlussfolgerungen aus Behauptungen (Axiomen) ableiten
- Kann Inkonsistenzen aufdecken (widersprüchliche Axiome identifizieren)
- Kann implizit enthaltene Informationen explizit machen („Logikrätsel lösen“)

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- $\exists$ *Items*: u. a. zu jedem Wikipedia-Eintrag:

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- $\exists$ *Items*: u. a. zu jedem Wikipedia-Eintrag:
  - `https://www.wikidata.org/wiki/Q252446` *Anif bei Salzburg*
  - `https://www.wikidata.org/wiki/Q4917288` *Control Engineering*

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- ∃ *Items*: u. a. zu jedem Wikipedia-Eintrag:
  - `https://www.wikidata.org/wiki/Q252446` *Anif bei Salzburg*
  - `https://www.wikidata.org/wiki/Q4917288` *Control Engineering*
- ∃ *Properties*:
  - `https://www.wikidata.org/wiki/P31` *is instance of*
  - `https://www.wikidata.org/wiki/P2534` *has definig formula*
- ∃ Statements (Kanten im Wissensgraph)

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- ∃ *Items*: u. a. zu jedem Wikipedia-Eintrag:
  - `https://www.wikidata.org/wiki/Q252446` *Anif bei Salzburg*
  - `https://www.wikidata.org/wiki/Q4917288` *Control Engineering*
- ∃ *Properties*:
  - `https://www.wikidata.org/wiki/P31` *is instance of*
  - `https://www.wikidata.org/wiki/P2534` *has definig formula*
- ∃ Statements (Kanten im Wissensgraph)
- Abfrageschnittstelle über SPARQL

https://query.wikidata.org/#SELECT%20 %3Fitem %3FitemLabel %3Fformula%0AWHERE %0A{%0A

Wikidata Query Service | Examples | Query Builder | Help | More tools | English

```
SELECT  ?item ?itemLabel ?formula
WHERE
{
  ?item wdt:P31 wd:Q877802.   # P31 → instance of; Q877802 → integral transformation
  ?item wdt:P2534 ?formula.   # P2534 → definig formula
  ?item rdfs:label ?itemLabel.
}
```

9 results in 246 ms | Code | Download | Link

item	itemLabel	formula
wd:Q2867	wavelet transform	$w(f)(t,a,b) = \int_{-\infty}^{\infty} f\left(\frac{x-b}{a}\right)\psi(x,a,b)\,dx$
wd:Q199691	Laplace transform	$F(s) = \int_0^{\infty} f(t)e^{-st}\,dt$
wd:Q210857	convolution	$(f*g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau = \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau$

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- ∃ *Items*: u. a. zu jedem Wikipedia-Eintrag:
  - `https://www.wikidata.org/wiki/Q252446` *Anif bei Salzburg*
  - `https://www.wikidata.org/wiki/Q4917288` *Control Engineering*
- ∃ *Properties*:
  - `https://www.wikidata.org/wiki/P31` *is instance of*
  - `https://www.wikidata.org/wiki/P2534` *has definig formula*
- ∃ Statements (Kanten im Wissensgraph)
- Abfrageschnittstelle über SPARQL

→ Repräsentation von mathematischen Inhalten in WD: umfangreich

↯ Repräsentation von regelungstheoretischen Inhalten in WD: **dürftig**

Thesen

These 1

Die gegenwärtig dominierenden Medien (Fließtext, Formeln, Grafiken) der Wissensrepräsentation sind in Bezug auf die angesprochenen Probleme suboptimal.

These 2

Formale Wissensrepräsentationsmethoden stellen eine vielversprechende Ergänzung dar.

These 3

Bisher existierende technische Lösungen (OWL, Wikidata, ...) sind ungeeignet.

Gliederung

☑ Motivation und Rückblick
☑ Formale Wissensrepräsentation: Ist-Stand
→ **Ansatz: Emergent Representation of Knowledge (ERK)**
☐ Ontology of Control Systems Engineering (OCSE)
☐ Zusammenfassung und Diskussion

Experimenteller Ansatz: ERK

Verklemmung:

(1) Kein geeigneter Repräsentationsformalismus → (2) Keine Inhalte → (3) Keine Anwendungen → (4) Keine Aufmerksamkeit → (5) Keine Entwicklung → (1)

Experimenteller Ansatz: ERK

Verklemmung:

① Kein geeigneter Repräsentationsformalismus → ② Keine Inhalte → ③ Keine Anwendungen → ④ Keine Aufmerksamkeit → ⑤ Keine Entwicklung → ①

Pragmatischer Ansatz:

① Repräsentationsformalismus: **E**mergent **R**epresentation of **K**nowledge

② initiale Inhalte: OCSE

Experimenteller Ansatz: ERK

Verklemmung:

① Kein geeigneter Repräsentationsformalismus → ② Keine Inhalte → ③ Keine Anwendungen → ④ Keine Aufmerksamkeit → ⑤ Keine Entwicklung → ①

Pragmatischer Ansatz:

① Repräsentationsformalismus: **E**mergent **R**epresentation of **K**nowledge

② initiale Inhalte: OCSE

Anforderungen

- Fokus auf Ausdrucksstärke
- Basiert auf menschenlesbarem Text
- Unterstützung für Automatisierung

Experimenteller Ansatz: ERK

Verklemmung:

① Kein geeigneter Repräsentationsformalismus → ② Keine Inhalte → ③ Keine Anwendungen → ④ Keine Aufmerksamkeit → ⑤ Keine Entwicklung → ①

Pragmatischer Ansatz:

① Repräsentationsformalismus: **E**mergent **R**epresentation of **K**nowledge

② initiale Inhalte: OCSE

Anforderungen

- Fokus auf Ausdrucksstärke
- Basiert auf menschenlesbarem Text
- Unterstützung für Automatisierung

} Erfüllt von Allzweckprogrammiersprache z. B. Python

ERK: Imperative statt deklarative Wissensrepräsentation

Deklarativ

- Formuliert in *Beschreibungs*sprache (z.B. OWL (Turtle Syntax))

ERK: Imperative statt deklarative Wissensrepräsentation

Deklarativ

- Formuliert in *Beschreibungs*sprache (z.B. OWL (Turtle Syntax))

```
I3749 rdfs:label "Cayley-Hamilton theorem".
I3749 rdfs:comment "every square matrix is a root of its own char. poly.".
I3749 rdf:type :I15_implication_proposition.
```

ERK: Imperative statt deklarative Wissensrepräsentation

Deklarativ

- Formuliert in *Beschreibungs*sprache (z.B. OWL (Turtle Syntax))

```
I3749 rdfs:label "Cayley-Hamilton theorem".
I3749 rdfs:comment "every square matrix is a root of its own char. poly.".
I3749 rdf:type :I15_implication_proposition.
```

Imperativ

- Formuliert in *Programmier*sprache (z.B. Python)

```
I3749 = p.create_item(
    R1__has_label="Cayley-Hamilton theorem",
    R2__has_description="every square matrix is a root of its own char. poly.",
    R4__is_instance_of=p.I15["implication proposition"],
)
```

ERK: Imperative statt deklarative Wissensrepräsentation (2)

Vorteile

- Direkte Programm-interne Repräsentation (kein Parsen)
- Direkte Erweiterbarkeit (Plugins im Graphen)

ERK: Imperative statt deklarative Wissensrepräsentation (2)

Vorteile

- Direkte Programm-interne Repräsentation (kein Parsen)
- Direkte Erweiterbarkeit (Plugins im Graphen)

Anwendung: Geltungsbereiche (*scopes*)

- Gliederung einer Aussage in *Kontext-Etablierung*, *Prämisse*, *Behauptung*
- Gesamtaussage abhängig davon in welchem *scope* eine Teilausage steht
- Deklarativ: nur sehr aufwendig umsetzbar
- Imperativ: einfach umsetzbar (automatisches Erzeugen von Hilfsknoten/kanten)

ERK: Imperative statt deklarative Wissensrepräsentation (2)

Vorteile

- Direkte Programm-interne Repräsentation (kein Parsen)
- Direkte Erweiterbarkeit (Plugins im Graphen)

Anwendung: Geltungsbereiche (*scopes*)

- Gliederung einer Aussage in *Kontext-Etablierung*, *Prämisse*, *Behauptung*
- Gesamtaussage abhängig davon in welchem *scope* eine Teilausage steht
- Deklarativ: nur sehr aufwendig umsetzbar
- Imperativ: einfach umsetzbar (automatisches Erzeugen von Hilfsknoten/kanten)

```
I3749 = p.create_item(
    R1__has_label="Cayley-Hamilton theorem",
    R2__has_description="every square matrix is a root of its own char. poly.",
    R4__is_instance_of=p.I15["implication proposition"],
)
```

ERK - Beispiel: *Inhalt* des Satzes von Cayley-Hamilton

```
with I3749["Cayley-Hamilton theorem"].scope("context") as cm:
    cm.new_var(A=uq_instance_of(I9906["square matrix"]))
    cm.new_var(n=uq_instance_of(p.I39["positive integer"]))
```

ERK - Beispiel: *Inhalt* des Satzes von Cayley-Hamilton

```
with I3749["Cayley-Hamilton theorem"].scope("context") as cm:
    cm.new_var(A=uq_instance_of(I9906["square matrix"]))
    cm.new_var(n=uq_instance_of(p.I39["positive integer"]))

    cm.new_var(P=p.instance_of(I4240["matrix polynomial"]))
    cm.new_var(Z=p.instance_of(I9905["zero matrix"]))
```

ERK - Beispiel: *Inhalt* des Satzes von Cayley-Hamilton

```
with I3749["Cayley-Hamilton theorem"].scope("context") as cm:
    cm.new_var(A=uq_instance_of(I9906["square matrix"]))
    cm.new_var(n=uq_instance_of(p.I39["positive integer"]))

    cm.new_var(P=p.instance_of(I4240["matrix polynomial"]))
    cm.new_var(Z=p.instance_of(I9905["zero matrix"]))

    cm.new_rel(cm.A, R5938["has row number"], cm.n)
    cm.new_rel(cm.A, R5940["has characteristic polynomial"], cm.P)
    cm.new_rel(cm.Z, R5938["has row number"], cm.n)
    cm.new_rel(cm.Z, R5939["has column number"], cm.n)
```

ERK - Beispiel: *Inhalt* des Satzes von Cayley-Hamilton

```
with I3749["Cayley-Hamilton theorem"].scope("context") as cm:
    cm.new_var(A=uq_instance_of(I9906["square matrix"]))
    cm.new_var(n=uq_instance_of(p.I39["positive integer"]))

    cm.new_var(P=p.instance_of(I4240["matrix polynomial"]))
    cm.new_var(Z=p.instance_of(I9905["zero matrix"]))

    cm.new_rel(cm.A, R5938["has row number"], cm.n)
    cm.new_rel(cm.A, R5940["has characteristic polynomial"], cm.P)
    cm.new_rel(cm.Z, R5938["has row number"], cm.n)
    cm.new_rel(cm.Z, R5939["has column number"], cm.n)

with I3749["Cayley-Hamilton theorem"].scope("assertions") as cm:
    cm.new_equation(lhs=cm.P(cm.A), rhs=cm.Z)
```

$$P(\mathbf{A}) = \mathbf{Z} \quad \text{mit } \mathbf{Z} := \mathbf{0}$$

ERK-Inferenzsystem

Nachteile Imperativer Repräsentation

- ...
- Kein Einsatz existierender *Reasoner*

ERK-Inferenzsystem

Nachteile Imperativer Repräsentation

- ...
- Kein Einsatz existierender *Reasoner*

$\rightarrow$ Regelbasierte Inferenz

- Beispiel: Klassifikation von $\dot{x} = a\sin(x) + bx^2 + cx + u$
- *Eingangsaffinität* $\supset$ *Polynomialität* (a=0) $\supset$ *Linearität* (a,b=0)

ERK-Inferenzsystem

Nachteile Imperativer Repräsentation

- ...
- Kein Einsatz existierender *Reasoner*

→ Regelbasierte Inferenz

- Beispiel: Klassifikation von $\dot{x} = a\sin(x) + bx^2 + cx + u$
- *Eingangsaffinität* $\supset$ *Polynomialität* (a=0) $\supset$ *Linearität* (a,b=0)
- Im Graph: `I4761["linear"]` `R17["is subproperty of"]` `I5247["polynomial"]`
- Wunsch: Schlussfolgerung von `I4761["linear"]` `R17` `I6091["control affine"]`

ERK-Inferenzsystem

Nachteile Imperativer Repräsentation

- ...
- Kein Einsatz existierender *Reasoner*

→ Regelbasierte Inferenz

- Beispiel: Klassifikation von $\dot{x} = a\sin(x) + bx^2 + cx + u$
- *Eingangsaffinität* ⊃ *Polynomialität* (a=0) ⊃ *Linearität* (a,b=0)
- Im Graph: `I4761["linear"]` `R17["is subproperty of"]` `I5247["polynomial"]`
- Wunsch: Schlussfolgerung von `I4761["linear"]` `R17` `I6091["control affine"]`
- Abstrakt: Transitivität der Relation `R17["is subproperty of"]`
- Wunsch: Regel soll selbst Teil des Wissensgraphen sein

ERK-Inferenzsystem: Regelspezifikation

```
I400 = p.create_item(
    R1__has_label="transitivity of R17__is_subproperty_of",
    R4__is_instance_of=p.I41["semantic rule"],
)
```

ERK-Inferenzsystem: Regelspezifikation

```
I400 = p.create_item(
    R1__has_label="transitivity of R17__is_subproperty_of",
    R4__is_instance_of=p.I41["semantic rule"],
)

with I400["subproperty rule 1"].scope("context") as cm:
    cm.new_var(P1=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P2=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P3=p.instance_of(p.I11["mathematical property"]))
```

ERK-Inferenzsystem: Regelspezifikation

```
I400 = p.create_item(
    R1__has_label="transitivity of R17__is_subproperty_of",
    R4__is_instance_of=p.I41["semantic rule"],
)

with I400["subproperty rule 1"].scope("context") as cm:
    cm.new_var(P1=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P2=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P3=p.instance_of(p.I11["mathematical property"]))

with I400.scope("premises") as cm:
    cm.new_rel(cm.P2, p.R17["is subproperty of"], cm.P1)
    cm.new_rel(cm.P3, p.R17["is subproperty of"], cm.P2)
```

ERK-Inferenzsystem: Regelspezifikation

```
I400 = p.create_item(
    R1__has_label="transitivity of R17__is_subproperty_of",
    R4__is_instance_of=p.I41["semantic rule"],
)

with I400["subproperty rule 1"].scope("context") as cm:
    cm.new_var(P1=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P2=p.instance_of(p.I11["mathematical property"]))
    cm.new_var(P3=p.instance_of(p.I11["mathematical property"]))

with I400.scope("premises") as cm:
    cm.new_rel(cm.P2, p.R17["is subproperty of"], cm.P1)
    cm.new_rel(cm.P3, p.R17["is subproperty of"], cm.P2)

with I400.scope("assertions") as cm:
    cm.new_rel(cm.P3, p.R17["is subproperty of"], cm.P1)
```

ERK-Inferenzsystem: Regelauswertung

- Für jede Regel: „Prototypgraphen“ konstruieren
  - lokale Variablen aus `scope("context")` → Knoten
  - Relationen aus `scope("premises")` → Kanten

ERK-Inferenzsystem: Regelauswertung

- Für jede Regel: „Prototypgraphen“ konstruieren
  - lokale Variablen aus `scope("context")` → Knoten
  - Relationen aus `scope("premises")` → Kanten

- Passende Knoten aus dem Gesamtgraph suchen
  - Mathematisches Problem: ***Subgraphisomorphismen*** finden
  - ∃ VF2-Algorithmus (fertig implementiert)

ERK-Inferenzsystem: Regelauswertung

- Für jede Regel: „Prototypgraphen“ konstruieren
  - lokale Variablen aus `scope("context")` → Knoten
  - Relationen aus `scope("premises")` → Kanten

- Passende Knoten aus dem Gesamtgraph suchen
  - Mathematisches Problem: ***Subgraphisomorphismen*** finden
  - ∃ VF2-Algorithmus (fertig implementiert)

- Beziehungen aus `scope("assertions")` abstrahieren und anwenden

ERK – Bemerkungen

- Alle *Item*s und `Relation`s haben eindeutige URI (z. B. `erk:/builtins#I41`)
- Unterstützung für Mehrsprachigkeit (Label, Beschreibung, …)
- Unterstützung für Qualifier (Kanten, die auf Kanten zeigen)
  - Semantische Information steckt in Knoten und Kanten
  - menschenlesbare Texte sind Hilfsattribute
  - Lesbarkeit für Mensch und Maschine: `I41["semantische Regel"@de]` $\hat{=}$ `I41`
- Zur Begriffswahl „Emergent“ (ERK)
  - *Emergenz* wörtlich: „das Auftauchen“
  - Bedeutung: Phänomen, dass in komplexen Systemen Eigenschaften auftreten, die nicht aus den Eigenschaften der Elemte vorhergesagt werden können.

  $\rightarrow$: *Das Ganze ist mehr als die Summe seiner Teile.*
  - Ursprünglich: ERK – „*Easy* Knowledge Representation“
- $\exists$ **RDF-Export $\rightarrow$ SPARQL Suche möglich**

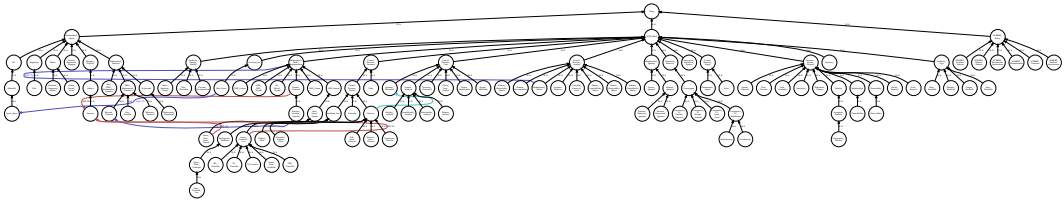
Gliederung

☑ Motivation und Rückblick
☑ Formale Wissensrepräsentation: Ist-Stand
☑ Ansatz: Emergent Representation of Knowledge (ERK)
→ **Ontology of Control Systems Engineering (OCSE)**
☐ Zusammenfassung und Diskussion

Ontology of Control Systems Engineering

2021: OCSE 0.1

- in OWL implementiert
- Taxonomie einfach umsetzbar
- Weitere Beziehungen schwierig (mangelnde OWL-Ausdrucksstärke)



Ontology of Control Systems Engineering

2021: OCSE 0.1

- in OWL implementiert
- Taxonomie einfach umsetzbar
- Weitere Beziehungen schwierig (mangelnde OWL-Ausdrucksstärke)

2022: OCSE 0.2

- Basierend auf ERK implementiert
- Größere Modellierungstiefe möglich

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? →

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? → Dynamische Systeme.

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? → Dynamische Systeme.

```
I5948 = p.create_item(
    R1__has_label="dynamical system",
    R4__is_instance_of=p.I2["Metaclass"] # <- Metaklassen-Instanzen sind Klassen
)
```

Geht es wirklich um Systeme?

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? → Dynamische Systeme.

```
I5948 = p.create_item(
    R1__has_label="dynamical system",
    R4__is_instance_of=p.I2["Metaclass"] # <- Metaklassen-Instanzen sind Klassen
)
```

Geht es wirklich um Systeme? → Es geht um *Modelle*.

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? → Dynamische Systeme.

```
I5948 = p.create_item(
    R1__has_label="dynamical system",
    R4__is_instance_of=p.I2["Metaclass"] # <- Metaklassen-Instanzen sind Klassen
)
```

Geht es wirklich um Systeme? → Es geht um *Modelle*.

```
I7641 = p.create_item(
    R1__has_label="general system model",
    R4__is_instance_of=p.I2["Metaclass"],
)
```

OCSE – Wie anfangen?

Worum geht es in der Regelungstheorie? → Dynamische Systeme.

```
I5948 = p.create_item(
    R1__has_label="dynamical system",
    R4__is_instance_of=p.I2["Metaclass"] # <- Metaklassen-Instanzen sind Klassen
)
```

Geht es wirklich um Systeme? → Es geht um *Modelle*.

```
I7641 = p.create_item(
    R1__has_label="general system model",
    R4__is_instance_of=p.I2["Metaclass"],
)
R7641 = p.create_relation(
    R1__has_label="has approximation",
    R8__has_domain_of_argument_1=I5948["dynamical system"],
    R11__has_range_of_result=I7641["general system model"],
)
```

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik
- Pragmatischer Ansatz: Irgendwo anfangen und schrittweise ergänzen

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik
- Pragmatischer Ansatz: Irgendwo anfangen und schrittweise ergänzen

Beispiel: `R8133["relative degree"]`
($\hat{=} \frac{d}{dt}$-Ordnung des Systemausgangs, die erstmals explizit vom Eingang abhängt)

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik
- Pragmatischer Ansatz: Irgendwo anfangen und schrittweise ergänzen

Beispiel: `R8133["relative degree"]`
($\hat{=} \frac{d}{dt}$-Ordnung des Systemausgangs, die erstmals explizit vom Eingang abhängt) benötigt:

- `I1371["iterated Lie derivative of scalar field"]`

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik
- Pragmatischer Ansatz: Irgendwo anfangen und schrittweise ergänzen

Beispiel: `R8133["relative degree"]`
($\hat{=} \frac{d}{dt}$-Ordnung des Systemausgangs, die erstmals explizit vom Eingang abhängt)
benötigt:

- `I1371["iterated Lie derivative of scalar field"]`
  - `I1347["Lie derivative of scalar field"]`

Lie Ableitung

$$\dot{x} = f(x) + g(x)u$$
$$y = h(x)$$

$$L_f h := \left(\frac{d}{dt} \varphi_t^f(x) \right) \Big|_{t=0}$$

OCSE – Repräsentation von Mathematik

- Regelungstheoretische Aussagen benötigen unverzichtbar Mathematik
- Pragmatischer Ansatz: Irgendwo anfangen und schrittweise ergänzen

Beispiel: `R8133["relative degree"]`
($\hat{=} \frac{d}{dt}$-Ordnung des Systemausgangs, die erstmals explizit vom Eingang abhängt)
benötigt:

- `I1371["iterated Lie derivative of scalar field"]`
  - `I1347["Lie derivative of scalar field"]`
    - `I2075["substitution"]`
    - `I3513["derivative w.r.t. scalar parameter"]`
    - `I2753["flow of a vector field"]`
    - `I9273["explicit first order ODE system"]`

...

Lie Ableitung

$$\dot{x} = f(x) + g(x)u$$
$$y = h(x)$$

$$L_f h := \left(\frac{d}{dt} \varphi_t^f(x) \right) \Big|_{t=0}$$

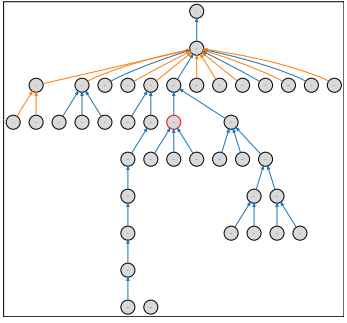
OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

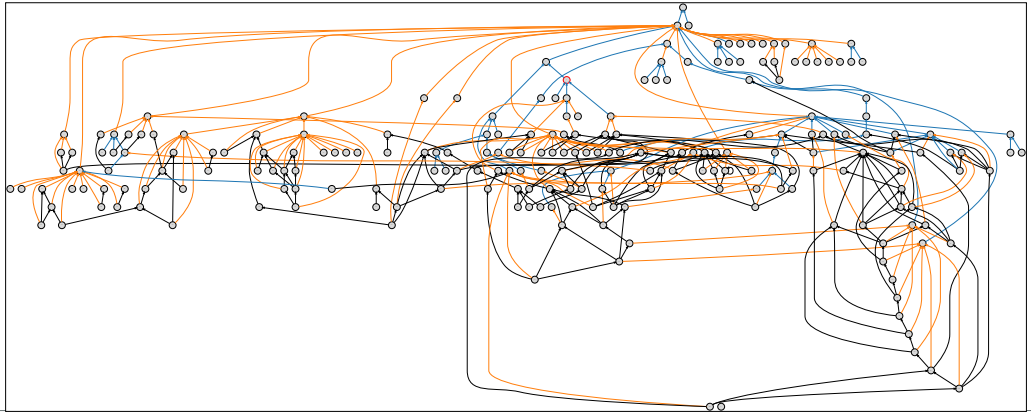
Visualisierung:



OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

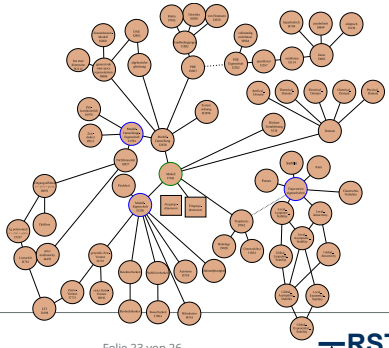
Visualisierung:



OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

Bisherige Anwendung:
Klassifikation von Enititäten im **A**utomatic **C**ontrol **K**nowledge **Rep**ository (ACKREP)
(Systemmodelle, Problembeschreibungen, Lösungsmethoden)

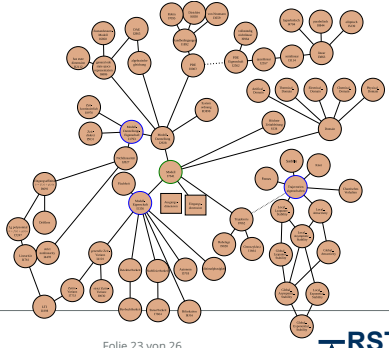


OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

Bisherige Anwendung:
Klassifikation von Enititäten im **A**utomatic **C**ontrol **K**nowledge **Rep**ository (ACKREP)
(Systemmodelle, Problembeschreibungen, Lösungsmethoden)

$\rightarrow$ Ermöglicht SPARQL-Suche
(z. B. exakt E-Z-linearisierbare Modelle mit $n > 3$)



OCSE – Status und Anwendungen

Status: `<erk:/builtins>` $\cup$ `<erk:/ocse/0.2>`: $\approx$ 230 Knoten, 400 Kanten

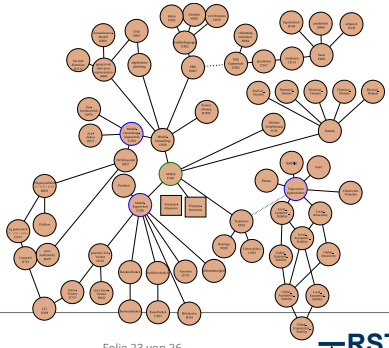
Bisherige Anwendung:
Klassifikation von Enititäten im **A**utomatic **C**ontrol **K**nowledge **Rep**ository (ACKREP)
(Systemmodelle, Problembeschreibungen, Lösungsmethoden)

$\rightarrow$ Ermöglicht SPARQL-Suche
(z. B. exakt E-Z-linearisierbare Modelle mit $n > 3$)

Mögliche zukünftige Anwendungen

- (Autom.) Klassifikation von Veröffentlichungen
  Denkbar: bis auf Satz- bzw. Gleichungsebene.
- Assistenzsoftware für Reglerentwurf

$\rightarrow$ Wissenstransfer



Gliederung

☑ Motivation und Rückblick
☑ Formale Wissensrepräsentation: Ist-Stand
☑ Ansatz: Emergent Representation of Knowledge (ERK)
☑ Ontology of Control Systems Engineering (OCSE)
→ **Zusammenfassung und Diskussion**

Zusammenfassung und Diskussion

- Formale Wissensrepräsentation potenziell nützlich für Wissenstransfer

Zusammenfassung und Diskussion

- Formale Wissensrepräsentation potenziell nützlich für Wissenstransfer
- Für Regelungstheorie existiert noch keine etablierte Technologie

Zusammenfassung und Diskussion

- Formale Wissensrepräsentation potenziell nützlich für Wissenstransfer
- Für Regelungstheorie existiert noch keine etablierte Technologie
- Experimenteller Vorschlag: ERK + OCSE (Code und Daten sind Open Source)

→ `https://ackrep.org`

Zusammenfassung und Diskussion

- Formale Wissensrepräsentation potenziell nützlich für Wissenstransfer
- Für Regelungstheorie existiert noch keine etablierte Technologie
- Experimenteller Vorschlag: ERK + OCSE (Code und Daten sind Open Source)

→ `https://ackrep.org`

Diskussion (offene Fragen)

- Grundsätzliche Tauglichkeit?
- (Automatisierte) Qualitätssicherung?
- Wissensintegration als sozialer Prozess?

Zusammenfassung und Diskussion

- Formale Wissensrepräsentation potenziell nützlich für Wissenstransfer
- Für Regelungstheorie existiert noch keine etablierte Technologie
- Experimenteller Vorschlag: ERK + OCSE (Code und Daten sind Open Source)

→ `https://ackrep.org`

Diskussion (offene Fragen)

- Grundsätzliche Tauglichkeit?
- (Automatisierte) Qualitätssicherung?
- Wissensintegration als sozialer Prozess?

→ Herzliche Einladung zur Kollaboration

Carsten.Knoll@tu-dresden.de

Ergänzungsfolien

Wissensrepräsentation (5): Open Research Knowledge Graph

- Projekt der Technischen Informationsbibliothek (TIB) Hannover
- Selbstbeschreibung: *Infrastruktur-Dienst zur Sammlung von akademischem Wissen in maschinenverarbeitbarer Form.*

Wissensrepräsentation (5): Open Research Knowledge Graph

- Projekt der Technischen Informationsbibliothek (TIB) Hannover
- Selbstbeschreibung: *Infrastruktur-Dienst zur Sammlung von akademischem Wissen in maschinenverarbeitbarer Form.*

Grundlegendes Schema:

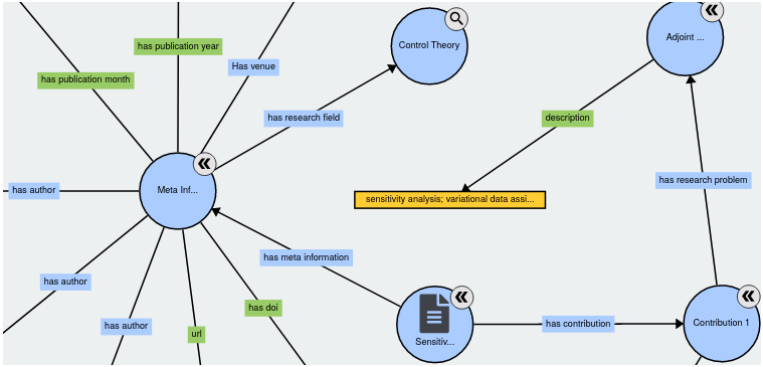
```
├── paper1
│   ├── contribution1
│   │   ├── research problem
│   │   │   └── ...
│   │   ├── method
│   │   │   └── ...
│   │   └── result
│   │       └── ...
│   └── ...
└ ...
```

Wissensrepräsentation (5): Open Research Knowledge Graph

- Projekt der Technischen Informationsbibliothek (TIB) Hannover
- Selbstbeschreibung: *Infrastruktur-Dienst zur Sammlung von akademischem Wissen in maschinenverarbeitbarer Form.*

Grundlegendes Schema:

```
├── paper1
│   ├── contribution1
│   │   ├── research problem
│   │   │   └── ...
│   │   ├── method
│   │   │   └── ...
│   │   └── result
│   │       └── ...
│   └── ...
└ ...
```



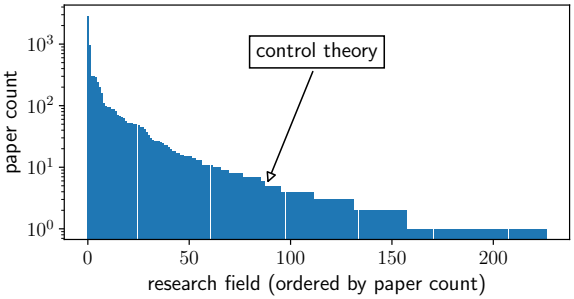
Wissensrepräsentation (5): Open Research Knowledge Graph

- Projekt der Technischen Informationsbibliothek (TIB) Hannover
- Selbstbeschreibung: *Infrastruktur-Dienst zur Sammlung von akademischem Wissen in maschinenverarbeitbarer Form.*

Grundlegendes Schema:

```
├── paper1
│   ├── contribution1
│   │   ├── research problem
│   │   │   └── ...
│   │   ├── method
│   │   │   └── ...
│   │   └── result
│   │       └── ...
│   └── ...
└ ...
```

Repräsentanz der Regelungstheorie:



Wissensrepräsentation (5): Open Research Knowledge Graph

- Projekt der Technischen Informationsbibliothek (TIB) Hannover
- Selbstbeschreibung: *Infrastruktur-Dienst zur Sammlung von akademischem Wissen in maschinenverarbeitbarer Form.*

Grundlegendes Schema:

```
├── paper1
│   ├── contribution1
│   │   ├── research problem
│   │   │   └── ...
│   │   ├── method
│   │   │   └── ...
│   │   └── result
│   │       └── ...
│   └── ...
└ ...
```

→ Bisher nur „grobe“ Wissensrepräsentation möglich/üblich

Formale Wissensrepräsentation (4): Wikidata und SPARQL

- Weltweit größter frei zugänglicher Wissensgraph
- Kollaborativ erstellt, von Wikimedia Foundation organisiert
- $\exists$ *Items*: u. a. zu jedem Wikipedia-Eintrag:
  - `https://www.wikidata.org/wiki/Q252446` *Anif bei Salzburg*
  - `https://www.wikidata.org/wiki/Q4917288` *Control Engineering*
- $\exists$ *Properties*:
  - `https://www.wikidata.org/wiki/P31` *is instance of*
  - `https://www.wikidata.org/wiki/P2534` *has definig formula*
- $\exists$ Statements (Kanten im Wissensgraph)
- Abfrageschnittstelle über SPARQL

$\rightarrow$ Repräsentation von mathematischen Inhalten in WD: umfangreich

$\lightning$ Repräsentation von regelungstheoretischen Inhalten in WD: **dürftig**

Main page
Community portal
Project chat
Create a new Item
Recent changes
Random Item
Query Service
Nearby
Help
Donate

Lexicographical data
Create a new Lexeme
Recent changes
Random Lexeme

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Concept URI
Cite this page

PID controller (Q716829)

control loop mechanism used in control engineering
proportional-integral-derivative controller

edit

In more languages

Statements

subclass of	control engineering 0 references Q4917288 + add reference + add value	edit
described by source	Armenian Soviet Encyclopedia volume 4 page(s) 313 1 reference + add value	edit
schematic		edit