

PVO30 Textual Information Systems

Petr Sojka

Faculty of Informatics
Masaryk University, Brno

Spring 2013

I Information about the course

Basic info

Prerequisites and classification

Course syllabus

Literature

II Basic notions TIS

Basic notions and classification of information systems

Notions of (T)IS, PVO30 in the context of teaching at FI MU

(T)IS classification

Mini questionnaire

Information retrieval systems—classification

Classification and formalization of IRS

II Basic notions TIS

Basic notions and classification of information systems

Notions of (T)IS, PVO30 in the context of teaching at FI MU

(T)IS classification

Mini questionnaire

Information retrieval systems—classification

Classification and formalization of IRS

II Basic notions TIS

Basic notions and classification of information systems

Notions of (T)IS, PVO30 in the context of teaching at FI MU

(T)IS classification

Mini questionnaire

Information retrieval systems—classification

Classification and formalization of IRS

Part I

Information about the course PVO30

Basic info

Prerequisites and classification

Course syllabus

Literature

Introduction

- Petr Sojka, sojka@fi.muni.cz
- Consulting hours Spring 2013:
Wednesday 13:00–13:45
Tuesday 15:15–16:30
or write an email with other suggestions to meet.
- Room C523/522, fifth floor of block C, Botanická 68a.
- Course homepage: <http://www.fi.muni.cz/~sojka/PV030/>
- Seminar (Tue 12:00–12:50, C416 → B311).

Introduction

- Petr Sojka, sojka@fi.muni.cz
- Consulting hours Spring 2013:
Wednesday 13:00–13:45
Tuesday 15:15–16:30
or write an email with other suggestions to meet.
- Room C523/522, fifth floor of block C, Botanická 68a.
- Course homepage: <http://www.fi.muni.cz/~sojka/PV030/>
- Seminar (Tue 12:00–12:50, C416 → B311).

Introduction

- Petr Sojka, sojka@fi.muni.cz
- Consulting hours Spring 2013:
Wednesday 13:00–13:45
Tuesday 15:15–16:30
or write an email with other suggestions to meet.
- Room C523/522, fifth floor of block C, Botanická 68a.
- Course homepage: <http://www.fi.muni.cz/~sojka/PV030/>
- Seminar (Tue 12:00–12:50, C416 → B311).

Introduction

- Petr Sojka, sojka@fi.muni.cz
- Consulting hours Spring 2013:
Wednesday 13:00–13:45
Tuesday 15:15–16:30
or write an email with other suggestions to meet.
- Room C523/522, fifth floor of block C, Botanická 68a.
- Course homepage: <http://www.fi.muni.cz/~sojka/PV030/>
- Seminar (Tue 12:00–12:50, C416 → B311).

Introduction

- Petr Sojka, sojka@fi.muni.cz
- Consulting hours Spring 2013:
Wednesday 13:00–13:45
Tuesday 15:15–16:30
or write an email with other suggestions to meet.
- Room C523/522, fifth floor of block C, Botanická 68a.
- Course homepage: <http://www.fi.muni.cz/~sojka/PV030/>
- Seminar (Tue 12:00–12:50, C416 → B311).

Topics and classification of the course

Prerequisites:

It is expected the student having basic knowledge of the theory of automata and formal languages (IB005), elementary knowledge of theories of complexity, software programming and systems.

Classification:

There is a system of classification based on written mid-term (30 points) and final (70 points) exams. In addition, one can get additional premium points based on the activities during lectures. Classification scale (changes reserved) z/k/E/D/C/B/A correspond to obtaining 48/54/60/66/72/78/84 points.

Dates of [final] exams will be announced via IS.muni.cz (probably three terms).

Topics and classification of the course

Prerequisites:

It is expected the student having basic knowledge of the theory of automata and formal languages (IB005), elementary knowledge of theories of complexity, software programming and systems.

Classification:

There is a system of classification based on written mid-term (30 points) and final (70 points) exams. In addition, one can get additional premium points based on the activities during lectures. Classification scale (changes reserved) z/k/E/D/C/B/A correspond to obtaining 48/54/60/66/72/78/84 points.

Dates of [final] exams will be announced via IS.muni.cz (probably three terms).

Topics

My books focus on *timeless* truth.
D. E. Knuth, Brno, 1996

An emphasis will be given to the explanation of basic principles, algorithms and (software) design techniques, creation and implementation of textual information systems (TIS)—storage and information retrieval.

Topics

My books focus on *timeless* truth.
D. E. Knuth, Brno, 1996

An emphasis will be given to the explanation of basic principles, algorithms and (software) design techniques, creation and implementation of textual information systems (TIS)—storage and information retrieval.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus

- ① Basic notions. TIS (text information system). Classification of information systems. From texts to Watson.
- ② Searching in TIS. Searching and pattern matching classification and data structures. Algorithms of Knuth-Morris-Pratt, Aho-Corasick, reg. expr.
- ③ Algorithms of Boyer-Moore, Commentz-Walter, Buczilowski.
- ④ Theory of automata for searching. Classification of searching problems. Searching with errors.
- ⑤ Indexes. Indexing methods. Data structures for searching and indexing.
- ⑥ Google as an example of search and indexing engine. Pagerank. Signature methods.
- ⑦ Query languages and document models: Boolean, vector, probabilistic, MMM, Paice.

Syllabus (cont.)

- ⑧ *Data compression. Basic notions. Entropy.*
- ⑨ *Statistical methods.*
- ⑩ *Compression methods based on dictionary.*
- ⑪ *Syntactic methods. Context modeling. Language modeling. Corpora linguistics.*
- ⑫ *Spell checking. Filtering information channels. Document classification. Neural nets for text compression.*

Syllabus (cont.)

- ⑧ *Data compression. Basic notions. Entropy.*
- ⑨ *Statistical methods.*
- ⑩ *Compression methods based on dictionary.*
- ① *Syntactic methods. Context modeling. Language modeling. Corpora linguistics.*
- ② *Spell checking. Filtering information channels. Document classification. Neural nets for text compression.*

Syllabus (cont.)

- ⑧ Data compression. Basic notions. Entropy.
- ⑨ Statistical methods.
- ⑩ Compression methods based on dictionary.
- ① Syntactic methods. Context modeling. Language modeling. Corpora linguistics.
- ② Spell checking. Filtering information channels. Document classification. Neural nets for text compression.







Syllabus (cont.)

- ⑧ Data compression. Basic notions. Entropy.
- ⑨ Statistical methods.
- ⑩ Compression methods based on dictionary.
- ❶ Syntactic methods. Context modeling. Language modeling. Corpora linguistics.
- ❷ Spell checking. Filtering information channels. Document classification. Neural nets for text compression.







Syllabus (cont.)

- ⑧ Data compression. Basic notions. Entropy.
- ⑨ Statistical methods.
- ⑩ Compression methods based on dictionary.
- ❶ Syntactic methods. Context modeling. Language modeling. Corpora linguistics.
- ❷ Spell checking. Filtering information channels. Document classification. Neural nets for text compression.







Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching*, Second edition, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images*, Second edition, Morgan Kaufmann Publishers, 1998.







Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching, Second edition*, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images, Second edition*, Morgan Kaufmann Publishers, 1998.







Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching*, Second edition, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images*, Second edition, Morgan Kaufmann Publishers, 1998.







Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching, Second edition*, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images, Second edition*, Morgan Kaufmann Publishers, 1998.

Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching*, Second edition, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images*, Second edition, Morgan Kaufmann Publishers, 1998.

Textbooks

-  [MEL] Melichar, B.: *Textové informační systémy*, skripta ČVUT Praha, 2. vydání, 1996.
-  [POK] Pokorný, J., Snášel, V., Húsek D.: *Dokumentografické informační systémy*, Karolinum Praha, 1998.
-  [KOR] Korfhage, R. R.: *Information Storage and Retrieval*, Wiley Computer Publishing, 1997.
-  [SMY] Smyth, B.: *Computing Patterns in Strings*, Addison Wesley, 2003.
-  [KNU] Knuth, D. E.: *The Art of Computer Programming, Vol. 3, Sorting and Searching*, Second edition, 1998.
-  [WMB] Witten I. H., Moffat A., Bell T. C.: *Managing Gigabytes: compressing and indexing documents and images*, Second edition, Morgan Kaufmann Publishers, 1998.

Other study materials



[HEL] Held, G.: *Data and Image Compression, Tools and Techniques*, John Wiley & Sons, 4. vydání 1996.



[MEH] Melichar B., Holub J., A 6D Classification of Pattern Matching Problems, *Proceedings of The Prague Stringology Club Workshop '97*, Prague, July 7, CZ.



[G00] Brin S., Page, L.: *The anatomy of a Large-Scale Hypertextual Web Search Engine*. *WWW7/Computer Networks* 30(1-7): 107-117 (1998).
<http://dbpubs.stanford.edu:8090/pub/1998-8>



[MeM] Mehryar Mohri: *On Some Applications of Finite-State Automata Theory to Natural Language Processing*, *Natural Language Engineering*, 2(1):61-80, 1996.
<http://www.research.att.com/~mohri/cl1.ps.gz>

Other study materials



[HEL] Held, G.: *Data and Image Compression, Tools and Techniques*, John Wiley & Sons, 4. vydání 1996.



[MEH] Melichar B., Holub J., A 6D Classification of Pattern Matching Problems, *Proceedings of The Prague Stringology Club Workshop '97*, Prague, July 7, CZ.







[G00] Brin S., Page, L.: *The anatomy of a Large-Scale Hypertextual Web Search Engine*. *WWW7/Computer Networks* 30(1-7): 107-117 (1998).
<http://dbpubs.stanford.edu:8090/pub/1998-8>



[MeM] Mehryar Mohri: *On Some Applications of Finite-State Automata Theory to Natural Language Processing*, *Natural Language Engineering*, 2(1):61-80, 1996.
<http://www.research.att.com/~mohri/cl1.ps.gz>

Other study materials

-  [HEL] Held, G.: *Data and Image Compression, Tools and Techniques*, John Wiley & Sons, 4. vydání 1996.
-  [MEH] Melichar B., Holub J., A 6D Classification of Pattern Matching Problems, *Proceedings of The Prague Stringology Club Workshop '97*, Prague, July 7, CZ.
-  [G00] Brin S., Page, L.: The anatomy of a Large-Scale Hypertextual Web Search Engine. *WWW7/Computer Networks* 30(1-7): 107-117 (1998).
<http://dbpubs.stanford.edu:8090/pub/1998-8>
-  [MeM] Mehryar Mohri: On Some Applications of Finite-State Automata Theory to Natural Language Processing, *Natural Language Engineering*, 2(1):61-80, 1996.
<http://www.research.att.com/~mohri/cl1.ps.gz>

Other study materials



[HEL] Held, G.: *Data and Image Compression, Tools and Techniques*, John Wiley & Sons, 4. vydání 1996.



[MEH] Melichar B., Holub J., A 6D Classification of Pattern Matching Problems, *Proceedings of The Prague Stringology Club Workshop '97*, Prague, July 7, CZ.



[G00] Brin S., Page, L.: The anatomy of a Large-Scale Hypertextual Web Search Engine. *WWW7/Computer Networks* 30(1-7): 107-117 (1998).
<http://dbpubs.stanford.edu:8090/pub/1998-8>



[MeM] Mehryar Mohri: On Some Applications of Finite-State Automata Theory to Natural Language Processing, *Natural Language Engineering*, 2(1):61-80, 1996.
<http://www.research.att.com/~mohri/cl1.ps.gz>

Other study materials (cont.)



[Sch] Schmidhuber J.: *Sequential neural text compression*, *IEEE Transactions on Neural Networks* 7(1), 142–146, 1996,
<http://www.idsia.ch/~juergen/onlinepub.html>



[SBA] Salton G., Buckley Ch., Allan J.: *Automatic structuring of text files*, *Electronic Publishing* 5(1), p. 1–17 (March 1992).
<http://columbus.cs.nott.ac.uk/compsci/epo/epodd/ep056gs.htm>



[WWW] *web pages of the course ~sojka/PV030/, DIS seminars*
<http://www.inf.upol.cz/dis>, <http://nlp.fi.muni.cz/>, *The Prague Stringology Club Workshop 1996–2008*
<http://cs.felk.cvut.cz/psc/>



Jones, S. K., Willett: *Readings in Information Retrieval*, Morgan Kaufman Publishers, 1997.

Other study materials (cont.)



[Sch] Schmidhuber J.: *Sequential neural text compression*, *IEEE Transactions on Neural Networks* 7(1), 142–146, 1996,
<http://www.idsia.ch/~juergen/onlinepub.html>



[SBA] Salton G., Buckley Ch., Allan J.: *Automatic structuring of text files*, *Electronic Publishing* 5(1), p. 1–17 (March 1992).
<http://columbus.cs.nott.ac.uk/compsci/epo/epodd/ep056gs.htm>



[WWW] web pages of the course ~sojka/PV030/, DIS seminars
<http://www.inf.upol.cz/dis>, <http://nlp.fi.muni.cz/>, The Prague Stringology Club Workshop 1996–2008
<http://cs.felk.cvut.cz/psc/>



Jones, S. K., Willett: *Readings in Information Retrieval*, Morgan Kaufman Publishers, 1997.

Other study materials (cont.)



[Sch] Schmidhuber J.: *Sequential neural text compression*, *IEEE Transactions on Neural Networks* 7(1), 142–146, 1996,
<http://www.idsia.ch/~juergen/onlinepub.html>



[SBA] Salton G., Buckley Ch., Allan J.: *Automatic structuring of text files*, *Electronic Publishing* 5(1), p. 1–17 (March 1992).
<http://columbus.cs.nott.ac.uk/compsci/epo/epodd/ep056gs.htm>






[WWW] web pages of the course ~sojka/PV030/, DIS seminars
<http://www.inf.upol.cz/dis>, <http://nlp.fi.muni.cz/>, The Prague Stringology Club Workshop 1996–2008
<http://cs.felk.cvut.cz/psc/>






Jones, S. K., Willett: *Readings in Information Retrieval*, Morgan Kaufman Publishers, 1997.




Other study materials (cont.)

-  Bell, T. C., Cleary, J. G., Witten, I. H.: *Text Compression*, Prentice Hall, Englewood Cliffs, N. J., 1991.
 -  Storer, J.: *Data Compression: Methods and Theory*, Computer Science Press, Rockville, 1988.
 -  journals *ACM Transactions on Information Systems*, *Theoretical Computer Science*, *Neural Network World*, *ACM Transactions on Computer Systems*, *Knowledge Acquisition*.
- knihovna.muni.cz, umarecka.cz (textbook Pokorný),

Other study materials (cont.)




-  Bell, T. C., Cleary, J. G., Witten, I. H.: *Text Compression*, Prentice Hall, Englewood Cliffs, N. J., 1991.
 -  Storer, J.: *Data Compression: Methods and Theory*, Computer Science Press, Rockville, 1988.
 -  journals *ACM Transactions on Information Systems*, *Theoretical Computer Science*, *Neural Network World*, *ACM Transactions on Computer Systems*, *Knowledge Acquisition*.
- knihovna.muni.cz, umarecka.cz (textbook Pokorný),

Other study materials (cont.)

-  Bell, T. C., Cleary, J. G., Witten, I. H.: *Text Compression*, Prentice Hall, Englewood Cliffs, N. J., 1991.
-  Storer, J.: *Data Compression: Methods and Theory*, Computer Science Press, Rockville, 1988.
-  journals *ACM Transactions on Information Systems*, *Theoretical Computer Science*, *Neural Network World*, *ACM Transactions on Computer Systems*, *Knowledge Acquisition*.

knihovna.muni.cz, umarecka.cz (textbook Pokorný),

Other study materials (cont.)

-  Bell, T. C., Cleary, J. G., Witten, I. H.: *Text Compression*, Prentice Hall, Englewood Cliffs, N. J., 1991.
-  Storer, J.: *Data Compression: Methods and Theory*, Computer Science Press, Rockville, 1988.
-  journals *ACM Transactions on Information Systems*, *Theoretical Computer Science*, *Neural Network World*, *ACM Transactions on Computer Systems*, *Knowledge Acquisition*.

knihovna.muni.cz, umarecka.cz (textbook Pokorný),

Part II

Basic notions of TIS

Basic notions and classification of information systems

Notions of (T)IS, PVO30 in the context of teaching at FI MU

(T)IS classification

Mini questionnaire

Information retrieval systems—classification

Classification and formalization of IRS

Basic notions and classification of information systems

Notions of (T)IS, PVO30 in the context of teaching at FI MU

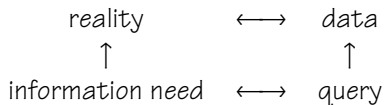
(T)IS classification

Mini questionnaire

Information retrieval systems—classification

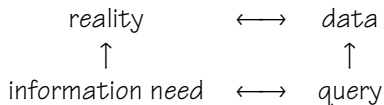
Classification and formalization of IRS

TIS—motivation



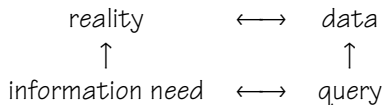
- Abstractions and mappings in information systems.
- Information needs about the reality—queries above data.
- Jeopardy game: Watson.

TIS—motivation



- ☞ Abstractions and mappings in information systems.
- ☞ Information needs about the reality—queries above data.
- ☞ Jeopardy game: Watson.

TIS—motivation



- ☞ Abstractions and mappings in information systems.
- ☞ Information needs about the reality—queries above data.
- ☞ Jeopardy game: Watson.

Notions of (T)IS

Definition: **Information system** is a system that allows purposeful arrangement of collection, storage, processing and delivering of information.

Definition: **Ectosystem** consists of IS users, investor of IS, and entrepreneur (user, funder, server). In the example of is.muni.cz they are users of IS, MU represented by bursar, and ICS and IS teams. Ectosystem is not under control of IS designer.

Definition: **Endosystem** consists of hardware used (media, devices), and software (algorithms, data structures) and is under control of IS designer.

Notions of (T)IS

Definition: **Information system** is a system that allows purposeful arrangement of collection, storage, processing and delivering of information.

Definition: **Ectosystem** consists of IS users, investor of IS, and entrepreneur (user, funder, server). In the example of is.muni.cz they are users of IS, MU represented by bursar, and ICS and IS teams. Ectosystem is not under control of IS designer.

Definition: **Endosystem** consists of hardware used (media, devices), and software (algorithms, data structures) and is under control of IS designer.

Notions of (T)IS

Definition: **Information system** is a system that allows purposeful arrangement of collection, storage, processing and delivering of information.

Definition: **Ectosystem** consists of IS users, investor of IS, and entrepreneur (user, funder, server). In the example of is.muni.cz they are users of IS, MU represented by bursar, and ICS and IS teams. Ectosystem is not under control of IS designer.

Definition: **Endosystem** consists of hardware used (media, devices), and software (algorithms, data structures) and is under control of IS designer.

Demands on TIS

- ☞ effectiveness (user)
- ☞ economics (funder)
- ☞ efficiency (server)

and from different preferences implied compromises. Our view will be view of TIS architect respecting requests of IS ecosystem. For topics related to ecosystem of IS see PVO45 Management IS.

Demands on TIS

- ☞ effectiveness (user)
- ☞ economics (funder)
- ☞ efficiency (server)

and from different preferences implied compromises. Our view will be view of TIS architect respecting requests of IS ecosystem. For topics related to ecosystem of IS see PVO45 Management IS.

From data to wisdom

- Data: concrete representation of a message in a form of sequence of symbols of an alphabet.
- Information: reflection of the known or the expected substance of realities. An information depends on the intended subject.
Viewpoints:
 - quantitative (information theory);
 - qualitative (meaning, semantics);
 - pragmatism (valuation: significance, usefulness, usability, periodicity, up-to-dateness, credibility);
 - the others (promptness, particularity, completeness, univocality, availability, costs of obtaining).
- Knowledge (znalost).
- Wisdom (moudrost).

From data to wisdom

- Data: concrete representation of a message in a form of sequence of symbols of an alphabet.
- Information: reflection of the known or the expected substance of realities. An information depends on the intended subject.
Viewpoints:
 - quantitative (information theory);
 - qualitative (meaning, semantics);
 - pragmatism (valuation: significance, usefulness, usability, periodicity, up-to-dateness, credibility);
 - the others (promptness, particularity, completeness, univocality, availability, costs of obtaining).
- Knowledge (znalost).
- Wisdom (moudrost).

From data to wisdom

- Data: concrete representation of a message in a form of sequence of symbols of an alphabet.
- Information: reflection of the known or the expected substance of realities. An information depends on the intended subject.
Viewpoints:
 - quantitative (information theory);
 - qualitative (meaning, semantics);
 - pragmatism (valuation: significance, usefulness, usability, periodicity, up-to-dateness, credibility);
 - the others (promptness, particularity, completeness, univocality, availability, costs of obtaining).
- Knowledge (znalost).
- Wisdom (moudrost).

From data to wisdom

- Data: concrete representation of a message in a form of sequence of symbols of an alphabet.
- Information: reflection of the known or the expected substance of realities. An information depends on the intended subject.
Viewpoints:
 - quantitative (information theory);
 - qualitative (meaning, semantics);
 - pragmatism (valuation: significance, usefulness, usability, periodicity, up-to-dateness, credibility);
 - the others (promptness, particularity, completeness, univocality, availability, costs of obtaining).
- Knowledge (znalost).
- Wisdom (moudrost).

Information process

Definition: **Information process** is a process of formation of information, its representation in a form of data, its processing, providing, and use. Operations with information correspond to this process.

Data/signals → Information → Knowledge → Wisdom.

Information process

Definition: **Information process** is a process of formation of information, its representation in a form of data, its processing, providing, and use. Operations with information correspond to this process.

Data/signals → Information → Knowledge → Wisdom.

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PVO03, PVO55, PV136, PB114).
- ③ Management information systems (PVO45).
- ④ Decision support systems (PVO98).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PVO03, PVO55, PV136, PB114).
- ③ Management information systems (PVO45).
- ④ Decision support systems (PVO98).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PVO03, PVO55, PV136, PB114).
- ③ Management information systems (PVO45).
- ④ Decision support systems (PVO98).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PV003, PV055, PV136, PB114).
- ③ Management information systems (PV045).
- ④ Decision support systems (PV098).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PV003, PV055, PV136, PB114).
- ③ Management information systems (PV045).
- ④ Decision support systems (PV098).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function

- ① Information retrieval systems.
- ② Database management systems (DBMS), relational DB (PB154, PB155, PV003, PV055, PV136, PB114).
- ③ Management information systems (PV045).
- ④ Decision support systems (PV098).
- ⑤ Expert systems, question answering systems, knowledge-based systems (PA031).
- ⑥ Information service systems (web 2.0).

IS classification by the prevailing function (cont.)

- ⑥ Specific information systems (*geographical* PV019, PA049, PA050, *medical* PV048, *environmental* PV044, *corporate* PV043, *state administration* PV058, PV059, *librarian* PV070); and also PV063. Application of database systems.

Related fields taught in FI:

Software engineering (PA102, PA105).

Similarity searching in multimedia data (PA128).

Efficient use of database systems (PA152).

Introduction to information retrieval (PV211).

IS classification by the prevailing function (cont.)

- ⑥ Specific information systems (*geographical* PV019, PA049, PA050, *medical* PV048, *environmental* PV044, *corporate* PV043, *state administration* PV058, PV059, *librarian* PV070); and also PV063. Application of database systems.

Related fields taught in FI:

Software engineering (PA102, PA105).

Similarity searching in multimedia data (PA128).

Efficient use of database systems (PA152).

Introduction to information retrieval (PV211).

IS classification by the prevailing function (cont.)

- ⑥ Specific information systems (*geographical* PV019, PA049, PA050, *medical* PV048, *environmental* PV044, *corporate* PV043, *state administration* PV058, PV059, *librarian* PV070); and also PV063. Application of database systems.

Related fields taught in FI:

Software engineering (PA102, PA105).

Similarity searching in multimedia data (PA128).

Efficient use of database systems (PA152).

Introduction to information retrieval (PV211).

IS classification by the prevailing function (cont.)

- ⑥ Specific information systems (geographical PV019, PA049, PA050, medical PV048, environmental PV044, corporate PV043, state administration PV058, PV059, librarian PV070); and also PV063. Application of database systems.

Related fields taught in FI:

Software engineering (PA102, PA105).

Similarity searching in multimedia data (PA128).

Efficient use of database systems (PA152).

Introduction to information retrieval (PV211).

IS classification by the prevailing function (cont.)

- ⑥ Specific information systems (geographical PV019, PA049, PA050, medical PV048, environmental PV044, corporate PV043, state administration PV058, PV059, librarian PV070); and also PV063. Application of database systems.

Related fields taught in FI:

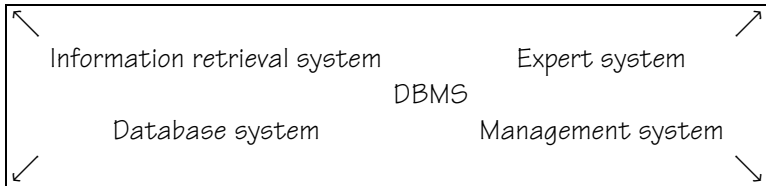
Software engineering (PA102, PA105).

Similarity searching in multimedia data (PA128).

Efficient use of database systems (PA152).

Introduction to information retrieval (PV211).

Diversity of TIS perspectives



Mini questionnaire

- ① What do you expect from this course? What was your motivation to enroll? Is the planned syllabus fine? Any changes or surprises?
- ② What do you not expect (you would rather eliminate)?
- ③ Which related courses have you already passed?
- ④ Practising IS usage (as a user)
 - a) Which (T)IS do you use?
 - b) Intensity? Frequency? How many searching per month?
 - c) Are you satisfied with it?
- ⑤ IS creation (server)
 - a) Which (T)IS and its component have you realized? Area, size?
 - b) Are you satisfied with it? Bottlenecks?

Mini questionnaire

- ① What do you expect from this course? What was your motivation to enroll? Is the planned syllabus fine? Any changes or surprises?
- ② What do you not expect (you would rather eliminate)?
- ③ Which related courses have you already passed?
- ④ Practising IS usage (as a user)
 - a) Which (T)IS do you use?
 - b) Intensity? Frequency? How many searching per month?
 - c) Are you satisfied with it?
- ⑤ IS creation (server)
 - a) Which (T)IS and its component have you realized? Area, size?
 - b) Are you satisfied with it? Bottlenecks?

Mini questionnaire

- ① What do you expect from this course? What was your motivation to enroll? Is the planned syllabus fine? Any changes or surprises?
- ② What do you not expect (you would rather eliminate)?
- ③ Which related courses have you already passed?
- ④ Practising IS usage (as a user)
 - a) Which (T)IS do you use?
 - b) Intensity? Frequency? How many searching per month?
 - c) Are you satisfied with it?
- ⑤ IS creation (server)
 - a) Which (T)IS and its component have you realized? Area, size?
 - b) Are you satisfied with it? Bottlenecks?

Mini questionnaire

- ① What do you expect from this course? What was your motivation to enroll? Is the planned syllabus fine? Any changes or surprises?
- ② What do you not expect (you would rather eliminate)?
- ③ Which related courses have you already passed?
- ④ Practising IS usage (as a user)
 - a) Which (T)IS do you use?
 - b) Intensity? Frequency? How many searching per month?
 - c) Are you satisfied with it?
- ⑤ IS creation (server)
 - a) Which (T)IS and its component have you realized? Area, size?
 - b) Are you satisfied with it? Bottlenecks?

Mini questionnaire

- ① What do you expect from this course? What was your motivation to enroll? Is the planned syllabus fine? Any changes or surprises?
- ② What do you not expect (you would rather eliminate)?
- ③ Which related courses have you already passed?
- ④ Practising IS usage (as a user)
 - a) Which (T)IS do you use?
 - b) Intensity? Frequency? How many searching per month?
 - c) Are you satisfied with it?
- ⑤ IS creation (server)
 - a) Which (T)IS and its component have you realized? Area, size?
 - b) Are you satisfied with it? Bottlenecks?

Basic notions and classification of information systems

Notions of (T)IS, PVO3O in the context of teaching at FI MU

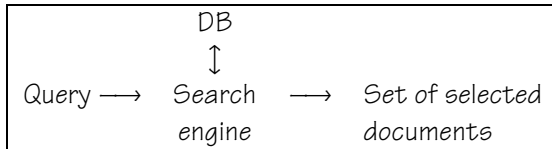
(T)IS classification

Mini questionnaire

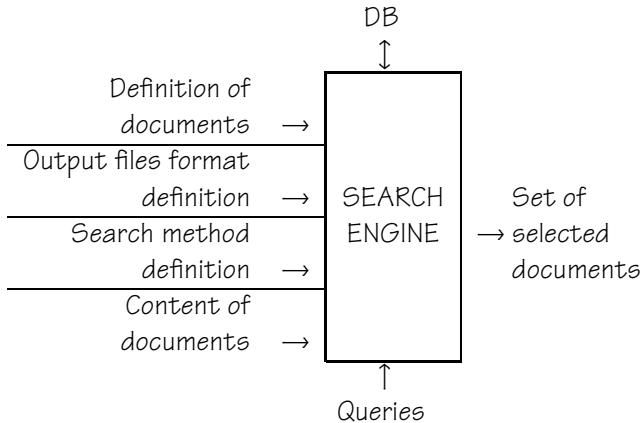
Information retrieval systems—classification

Classification and formalization of IRS

Information retrieval systems (IRS)—principles



An empty IRS



Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow an element. Indexing of elements by natural numbers. Not necessarily numbers, but labels.

- 0) Every element has unique label.
- 1) Every labeled element x (except for the leftmost one) has a clear predecessor referred to as $pred(x)$.
- 2) Every labeled element x (except for the rightmost one) has a clear successor referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for the leftmost one) has a clear predecessor referred to as $pred(x)$.
- 2) Every labeled element x (except for the rightmost one) has a clear successor referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for the leftmost one) has a clear predecessor referred to as $pred(x)$.
- 2) Every labeled element x (except for the rightmost one) has a clear successor referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem

Concatenation: string of beads. A bead \rightarrow **an element**. Indexing of elements by natural numbers. Not necessarily numbers, but **labels**.

- 0) Every element has unique label.
- 1) Every labeled element x (except for **the leftmost one**) has a clear **predecessor** referred to as $pred(x)$.
- 2) Every labeled element x (except for **the rightmost one**) has a clear **successor** referred to as $succ(x)$.
- 3) If the element x is not the leftmost one,
 $x = succ(pred(x))$.
- 4) If the element x is not the rightmost one,
 $x = pred(succ(x))$.
- 5) For every two different elements x and y , there exists a positive number k that is either $x = succ^k(y)$ or $x = pred^k(y)$.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: *a string* is a set of elements which meets the rules 0)–5).

Definition: *a linear string*: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: *a necklace*.

Definition: *an alphabet A. Letters of the alphabet, A^+ . An empty string ϵ .*

Definition: *a finite chain $A^* = A^+ \cup \{\epsilon\}$.*

Definition: *a linear string over A*: a member of A^+ .

Definition: *a pattern. A text.*

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: *a string* is a set of elements which meets the rules 0)–5).

Definition: *a linear string*: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: *a necklace*.

Definition: *an alphabet A. Letters of the alphabet, A^+ . An empty string ϵ .*

Definition: *a finite chain $A^* = A^+ \cup \{\epsilon\}$.*

Definition: *a linear string over A: a member of A^+ .*

Definition: *a pattern. A text.*

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . *Letters of the alphabet*. A^+ . *An empty string* ϵ .

Definition: **a finite chain** $A^* = A^+ \cup \{\epsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. *A text*.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: *an alphabet A. Letters of the alphabet. A^+ . An empty string ϵ .*

Definition: *a finite chain $A^* = A^+ \cup \{\epsilon\}$.*

Definition: *a linear string over A: a member of A^+ .*

Definition: *a pattern. A text.*

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet A** . *Letters of the alphabet. A^+ . An empty string ϵ .*

Definition: *a finite chain $A^* = A^+ \cup \{\epsilon\}$.*

Definition: *a linear string over A : a member of A^+ .*

Definition: *a pattern. A text.*

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ϵ .

Definition: **a finite chain** $A^* = A^+ \cup \{\epsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

Searching—formalization of the problem (cont.)

The *concatenation* term:

Definition: **a string** is a set of elements which meets the rules 0)–5).

Definition: **a linear string**: a string that has a finitely many elements including the leftmost and rightmost ones.

Definition: **a necklace**.

Definition: **an alphabet** A . **Letters of the alphabet**. A^+ . **An empty string** ε .

Definition: **a finite chain** $A^* = A^+ \cup \{\varepsilon\}$.

Definition: **a linear string over** A : a member of A^+ .

Definition: **a pattern**. **A text**.

IRS—classification

- ① Classification according to the passing direction:
left-to-right/right-to-left.
- ② Classification according to (pre)processing of the text and the pattern:
 - *ad fontes* (searching in the text itself);
 - text surrogate (searching in the substitution of the text);
 - substitutions:
 - an index: an ordered list of significant elements together with references to the original text;
 - a signature: a string of indicators that shows the occurrence of significant elements in the text.

IRS—classification

- ① Classification according to the passing direction:
left-to-right/right-to-left.
- ② Classification according to (pre)processing of the text and the pattern:
 - *ad fontes* (searching in the text itself);
 - *text surrogate* (searching in the substitution of the text);
 - *substitutions*:
 - an index: an ordered list of significant elements together with references to the original text;
 - a signature: a string of indicators that shows the occurrence of significant elements in the text.

IRS—classification

- ① Classification according to the passing direction:
left-to-right/right-to-left.
- ② Classification according to (pre)processing of the text and the pattern:
 - *ad fontes* (searching in the text itself);
 - text surrogate (searching in the substitution of the text);
 - *substitutions*:
 - an index: an ordered list of significant elements together with references to the original text;
 - a signature: a string of indicators that shows the occurrence of significant elements in the text.

IRS—classification

- ① Classification according to the passing direction:
left-to-right/right-to-left.
- ② Classification according to (pre)processing of the text and the pattern:
 - *ad fontes* (searching in the text itself);
 - text surrogate (searching in the substitution of the text);
 - substitutions:
 - an index: an ordered list of significant elements together with references to the original text;
 - a signature: a string of indicators that shows the occurrence of significant elements in the text.

IRS—classification (cont.)

	text preprocessing		
		no	yes
pattern preprocessing	no	I	III
	yes	II	IV

- I – elementary algorithms
- II – creating a search engine
- III – indexing methods
- IV – signature methods

Searching—the formulation of the problem

Classification according to the cardinality of the patterns' set:

- ① Search for a single pattern V in the text T . The result: yes/no.
- ② Search for a finite set of patterns $P = \{v_1, v_2, \dots, v_k\}$. The result: information about position of some of the entered patterns.
- ③ Search for an infinite set of patterns assigned by a regular expression R . R defines a potentially infinite set $L(R)$. The result: information about position of some of the patterns from $L(R)$.

Alternatives to the formulation of the searching problem:

- a) the first occurrence;
- b) the all occurrences without overlapping;
- c) the all occurrences including overlapping.

Searching—the formulation of the problem

Classification according to the cardinality of the patterns' set:

- ① Search for a single pattern V in the text T . The result: yes/no.
- ② Search for a finite set of patterns $P = \{v_1, v_2, \dots, v_k\}$. The result: information about position of some of the entered patterns.
- ③ Search for an infinite set of patterns assigned by a regular expression R . R defines a potentially infinite set $L(R)$. The result: information about position of some of the patterns from $L(R)$.

Alternatives to the formulation of the searching problem:

- a) the first occurrence;
- b) the all occurrences without overlapping;
- c) the all occurrences including overlapping.

Searching—the formulation of the problem

Classification according to the cardinality of the patterns' set:

- ① Search for a single pattern V in the text T . The result: yes/no.
- ② Search for a finite set of patterns $P = \{v_1, v_2, \dots, v_k\}$. The result: information about position of some of the entered patterns.
- ③ Search for an infinite set of patterns assigned by a regular expression R . R defines a potentially infinite set $L(R)$. The result: information about position of some of the patterns from $L(R)$.

Alternatives to the formulation of the searching problem:

- a) the first occurrence;
- b) the all occurrences without overlapping;
- c) the all occurrences including overlapping.

Searching—the formulation of the problem

Classification according to the cardinality of the patterns' set:

- ① Search for a single pattern V in the text T . The result: yes/no.
- ② Search for a finite set of patterns $P = \{v_1, v_2, \dots, v_k\}$. The result: information about position of some of the entered patterns.
- ③ Search for an infinite set of patterns assigned by a regular expression R . R defines a potentially infinite set $L(R)$. The result: information about position of some of the patterns from $L(R)$.

Alternatives to the formulation of the searching problem:

- a) the first occurrence;
- b) the all occurrences without overlapping;
- c) the all occurrences including overlapping.

Part III

Exact search

I. SE without preprocessing both patterns and the text
Rudimentary search algorithm

I. SE without preprocessing both patterns and the text
Rudimentary search algorithm

Naïve search, brute force search, rudimentary search algorithm

```
proc Brute-Force-Matcher(PATTERN, TEXT) :  
T:=length[TEXT]; P:=length[PATTERN];  
for i:=0 to T-P do  
    if PATTERN[1..P]=TEXT[i+1..i+P]  
    then print "The pattern was found at the position i.";
```

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Time complexity analysis of naïve search

- The complexity is measured by number of comparison, the length of a pattern P , the length of text T .
- The upper estimate $S = P \cdot (T - P + 1)$, thus $O(P \times T)$.
- The worst case $PATTERN = a^{P-1}b$, $TEXT = a^{T-1}b$.
- Natural languages: (average) complexity (number of comparison) substantially smaller, since the equality of prefixes doesn't occur very often. For English: $S = C_E \cdot (T - P + 1)$, C_E empirically measured 1.07, i. e. practically linear.
- C_{CZ} ? C_{CZ} vs. C_E ?
- Any speedups? An application of several patterns? An infinite number?
- We will see the version (S, Q, Q') of the algorithm in the seminar.

Naïve search—algorithms

Express the time complexity of the following search algorithms using the variables c and s , where c is the number of the tests and these statements are true:

- if the index i is found, then $c = i$ and $s = 1$;
- otherwise, $c = T$ and $s = 0$.

Naïve search—algorithm S

```

input:  var TEXT : array[1..T] of word;
        PATTERN : word;
output (in the variable FOUND): yes/no

1  I:=1;
c  while I ≤ T do
    begin
c      if TEXT[I]=PATTERN then break;
c-s   inc(I);
    end;
2  FOUND:=(I≤T);

```

On the left side, there is the time complexity of the statements.

And so the overall time complexity is $O(T) = 3c - s + 3$.

The maximum complexity (which is commonly stated) is $O(T) = 3T + 3$.

Algorithm Q or how about using the end stop/skid (zarážka)

```

input:  var TEXT : array[1..T+1] of word; PATTERN : word;
output (in the variable FOUND): yes/no

1  I:=1;
1  TEXT[T+1]:=PATTERN;
c  while TEXT[I]<>PATTERN do
c-1  inc(I);
2  FOUND:=(I<>T+1)

```

In this case, the index is always found; therefore it is stated on the last but one line of the algorithm that the complexity is $c - 1$ instead of $c - s$ (although they are equivalent). Furthermore, it is necessary to realize that the maximal possible value of c is greater by one than in the previous algorithm (stating $c + 1$ instead of c would not be correct, though). The overall complexity: $O(T) = 2c + 3$. The maximum complexity: $O(T) = 2T + 5$.

Algorithm Q' or how about using the cycle expansion

```

input: var TEXT : array[1..T+1] of word;
        PATTERN : word;
output (in the variable FOUND): yes/no

1      I:=1;
1      TEXT[T+1]:=PATTERN;
⌊c/2⌋  while TEXT[I]<>PATTERN do
        begin
⌊c/2⌋      if TEXT[I+1]=PATTERN then break;
⌊(c-1)/2⌋  I:=I+2;
        end;
3      FOUND:=(I<T) or (TEXT[T]=PATTERN);

```

The overall complexity: $O(T) = c + \lfloor (c-1)/2 \rfloor + 5$.

The maximum complexity: $O(T) = T + \lfloor T/2 \rfloor + 6$.

The condition at the end of the algorithm guarantees its functionality (however, it is not the only way of handling the cycle incrementation by two).