Chapter I:I (continued)

I. Style Guide

- ❑ Generic Hints
- ❑ Mathematical Notation
- ❑ Style Guide Latex
- ❑ Style Guide Adobe Illustrator
- ❑ Style Test

Style Test

Sizes

paperheight=496.93001pt

paperwidth=662.59087pt

baselineskip=23.0pt

parskip=9.19986pt

bsparskip=9.19986pt

leftmargin=39.6139pt

labelwidth=27.72974pt

Style Test

Listing

- V
- V
- V
  - w
  - w
    - x
    - x
      - y
      - y
    - x
  - w
- V

Style Test

Listing (continued)

1. v
2. v
3. v
   (a) w
   (b) w
      (i) x
      (ii) x
         A. y
         B. y
      (iii) x
   (c) w
4. v

Style Test

Listing (continued)

a) hello

- ❑ hello
  - – automatic text categorization
  - – automatic text categorization
- ❑ hello
  - – automatic text categorization
  - – automatic text categorization

1. hello

- ❑ hello
  - – automatic text categorization
  - – automatic text categorization
- ❑ hello
  - – automatic text categorization
  - – automatic text categorization

Style Test

Listing (continued)

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
18.
19.

Style Test

Skips: normalsize

line
newline

par

bspar1

bspar2

bspar3

bspar4

bspar5

Style Test

Skips: small

line
newline

par

bspar1

bspar2

bspar3

bspar4

bspar5

Style Test

Fonts

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

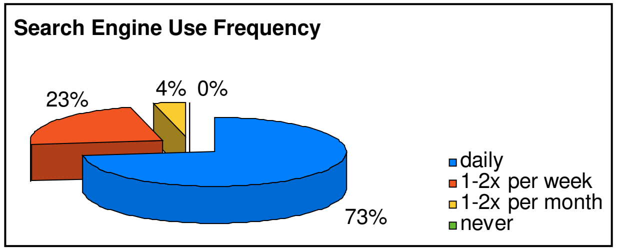
äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

äß **äß AB** $\mathbf{x}x * y = z$ $\mathbf{N}$ $\mathcal{C}$ $\models\notin$ $\pi\Pi$ `courier`**`fett`**

Style Test

Figures

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA



BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB

Style Test

Environments: Verbatim

```
\begin{bsslide}[Style Test]
\colortext{Algorithm}
\par
\renewcommand{\baselinestretch}{0.95}
\small\tt
BF$(s,\mathit{successors},\star,f)$
\begin{enumerate}
\renewcommand{\itemsep}{-2pt}
\item
$\mathit{insert}(s,\mathtt{OPEN})$;
\item
LOOP
\item
...
```

Style Test

Algorithm

```
BF($s$, $\textit{successors}$, $\star$, $f$)

  1.  $\textit{insert}(s, \texttt{OPEN})$;
  2.  LOOP
  3.    IF ($\texttt{OPEN} = \emptyset$) THEN RETURN($'\textit{Failure}'$);
  4.    $n = \textit{min}(\texttt{OPEN}, f)$;  // $n$ minimizes $f$ wrt. to all nodes in OPEN.
        $\textit{remove}(n, \texttt{OPEN})$;
        $\textit{push}(n, \texttt{CLOSED})$;  // Expanded nodes live here.
  5.    FOREACH $n'$ IN $\textit{successors}(n)$ DO
          $\textit{set\_backpointer}(n', n)$;
          IF $\star(n')$ THEN RETURN($n'$);
          $n'_{old} = \textit{node\_eq\_state}(n', \texttt{OPEN}, \texttt{CLOSED})$;
          IF ($n'_{old} = \texttt{NULL}$)
          THEN $\textit{insert}(n', \texttt{OPEN})$  // $n'$ encodes a new state.
          ELSE
            IF ($f(n') < f(n'_{old})$)
            THEN // The state of $n'$ can be reached via a cheaper path.
              $\textit{insert}(n', \texttt{OPEN})$;
              IF $\textit{member}(n'_{old}, \texttt{OPEN})$
              THEN $\textit{remove}(n'_{old}, \texttt{OPEN})$
              ELSE $\textit{remove}(n'_{old}, \texttt{CLOSED})$
            ENDIF
          ENDIF
        ENDDO
  6.  ENDLOOP
```

(1) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z (2) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G
H I J K L M N O P Q R S T U V W X Y Z (3) a b c d e f g h i j k l m n o p q r s t u v w
x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (4) a b c d e f g h i j k l
m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
(5) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z (6) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G
H I J K L M N O P Q R S T U V W X Y Z (7) a b c d e f g h i j k l m n o p q r s t u v w
x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (8) a b c d e f g h i j k l
m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
(9) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q
R S T U V W X Y Z (10) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G
H I J K L M N O P Q R S T U V W X Y Z (11) a b c d e f g h i j k l m n o p q r s t u v
w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (12) a b c d e f g h i j
k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
(13) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P
Q R S T U V W X Y Z (14) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F
G H I J K L M N O P Q R S T U V W X Y Z (15) a b c d e f g h i j k l m n o p q r s t u v
w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (16) a b c d e f g h i j
k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

(17) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (18) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (19) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (20) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

(1) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X
Y Z (2) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V
W X Y Z (3) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T
U V W X Y Z (4) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R
S T U V W X Y Z (5) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P
Q R S T U V W X Y Z (6) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N
O P Q R S T U V W X Y Z (7) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L
M N O P Q R S T U V W X Y Z (8) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J
K L M N O P Q R S T U V W X Y Z (9) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H
I J K L M N O P Q R S T U V W X Y Z (10) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F
G H I J K L M N O P Q R S T U V W X Y Z (11) a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D
E F G H I J K L M N O P Q R S T U V W X Y Z (12) a b c d e f g h i j k l m n o p q r s t u v w x y z A B
C D E F G H I J K L M N O P Q R S T U V W X Y Z (13) a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (14) a b c d e f g h i j k l m n o p q r s t u v w
x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (15) a b c d e f g h i j k l m n o p q r s t u
v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (16) a b c d e f g h i j k l m n o p q r
s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (17) a b c d e f g h i j k l m n o
p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (18) a b c d e f g h i j k l
m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (19) a b c d e f g h i
j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z (20) a b c d e f

g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Remarks:

- ❑ Finding structure in data objects is an ambiguous task, since structure that is to be found depends on a user's intentions, say, a user's information need. Take for example a text collection that contains financial and political news articles. When a user's task is to publish some of them within a newspaper, it might be sufficient to sort them into two categories, each of which is given to a responsible editor. On the other hand, an editor might wish that political articles are further subdivided into domestic and foreign-affairs articles.

- ❑ Given that a user's intention is clear, a data model for the objects has to be formed. Therefore, a function that maps the original objects onto abstract representations must be found. Then, a similarity measure must be derived, which numerically quantifies to which extent two of the object representations are related. Consequently, the data model along with the similarity measure determines how good a structure within the abstract representation can be detected at all.