Computer Science and `atlas-preamble`

Responsible for `compsci` preset: Tassilo Schwarz*

June 6, 2022

1 Introduction

To use computer science notation and packages in your document with `atlas-preamble`, use the `compsci` preset option:

```
\usepackage[compsci]{atlas-preamble}
```

2 Theoretical notation macros

The following often-used symbols in theoretical computer science get abbreviated macros:

`\Oh{}`	`\oh{}`	`\Th{}`	`\Om{}`	`\om{}`
$\mathcal{O}\left(\frac{n^2}{m}\right)$	$o\left(\frac{n^2}{m}\right)$	$\Theta\left(\frac{n^2}{m}\right)$	$\Omega\left(\frac{n^2}{m}\right)$	$\omega\left(\frac{n^2}{m}\right)$

3 Pseudocode

For the example algorithm using `algpseudocode` from the `algorithmicx` documentation, see Algorithm 1.

4 Complexity theoretic terms

The `complexity` package provides nice typesetting for complexity theoretic terms like P, NP, $\mathsf{EXPSPACE}$, etc. However, it redefines the following built-in commands as follows: L (originally Ł), P (originally ¶), and S (originally §). Access these using `\default{L,P,S}`, respectively.

*Email: `lastname@maths.ox.ac.uk`

Algorithm 1 The Bellman-Kalaba algorithm

Input: Some input **Output:** Some output

```
1: procedure BellmanKalaba(G, u, l, p)
2:     repeat  t times
3:         This will be repeated t times
4:     end
5:     ▷ This is one line only for a comment.
6:     for all v ∈ V(G) do
7:         l(v) ← ∞
8:     end for
9:     l(u) ← 0
10:    repeat
11:        for i ← 1, n do
12:            min ← l(v_i)
13:            for j ← 1, n do
14:                if min > e(v_i, v_j) + l(v_j) then
15:                    min ← e(v_i, v_j) + l(v_j)
16:                    p(i) ← v_j
17:                end if
18:            end for
19:            l'(i) ← min
20:        end for
21:        changed ← l ≠ l'
22:        l ← l'
23:    until ¬changed
24: end procedure

                                        ▷ Here we have extra space

25: procedure FindPathBK(v, u, p)
26:    if v = u then
27:        Write v
28:    else
29:        w ← v
30:        while w ≠ u do
31:            Write w
32:            w ← p(w)
33:        end while
34:    end if
35: end procedure
```