DELPHICON 2020

THE OFFICIAL ONLINE CONFERENCE ALL ABOUT EMBARCADERO DELPHI

HIGH PERFORMANCE DELPHI

Primož Gabrijelčič

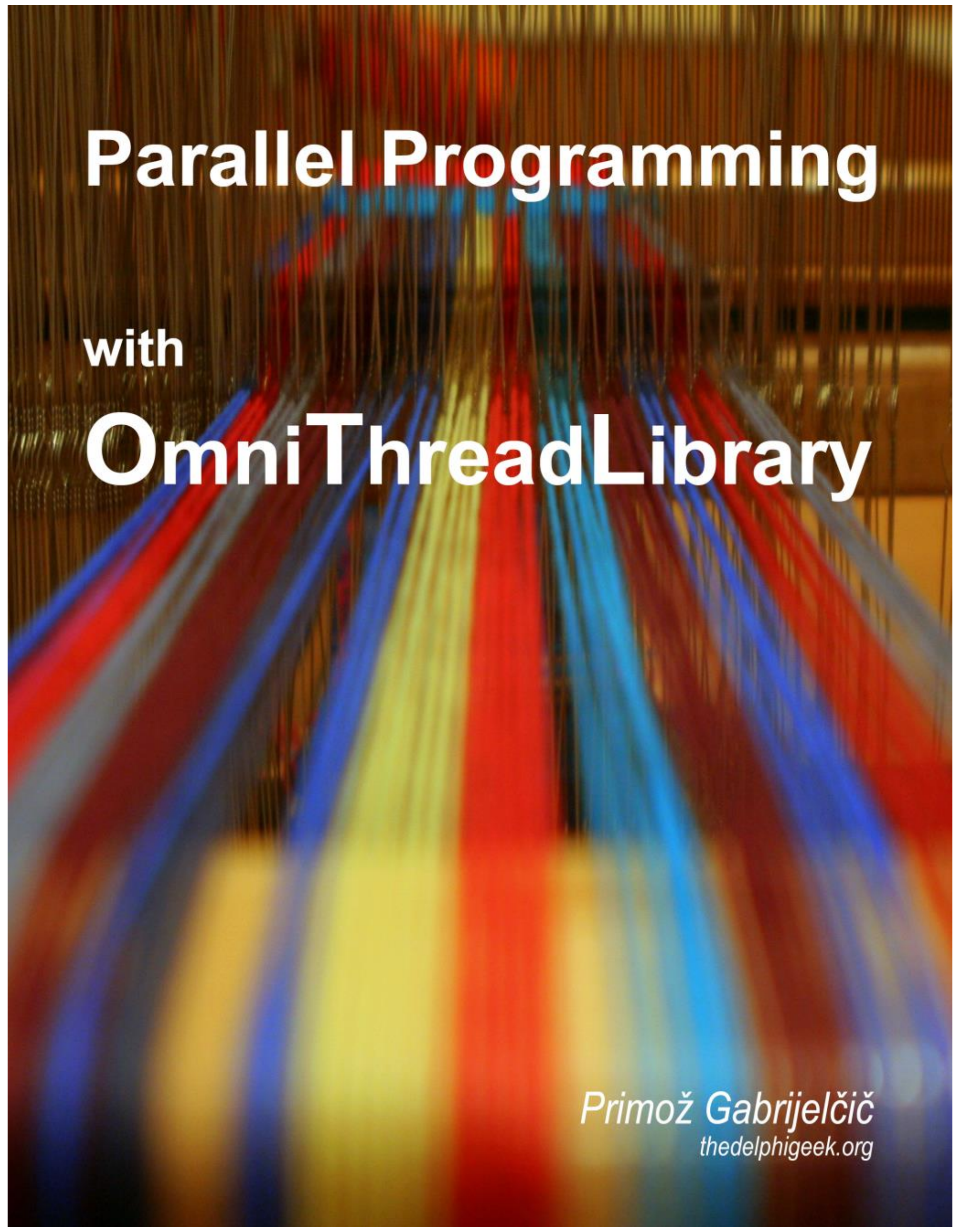
Primož Gabrijelčič

- programmer, MVP, writer, blogger, consultant, speaker

Blog	*http://thedelphigeek.com*
Twitter	*@thedelphigeek*
Skype	*gabr42*
LinkedIn	*gabr42*
GitHub	*gabr42*
SO	*gabr*

- http://primoz.gabrijelcic.org

Books



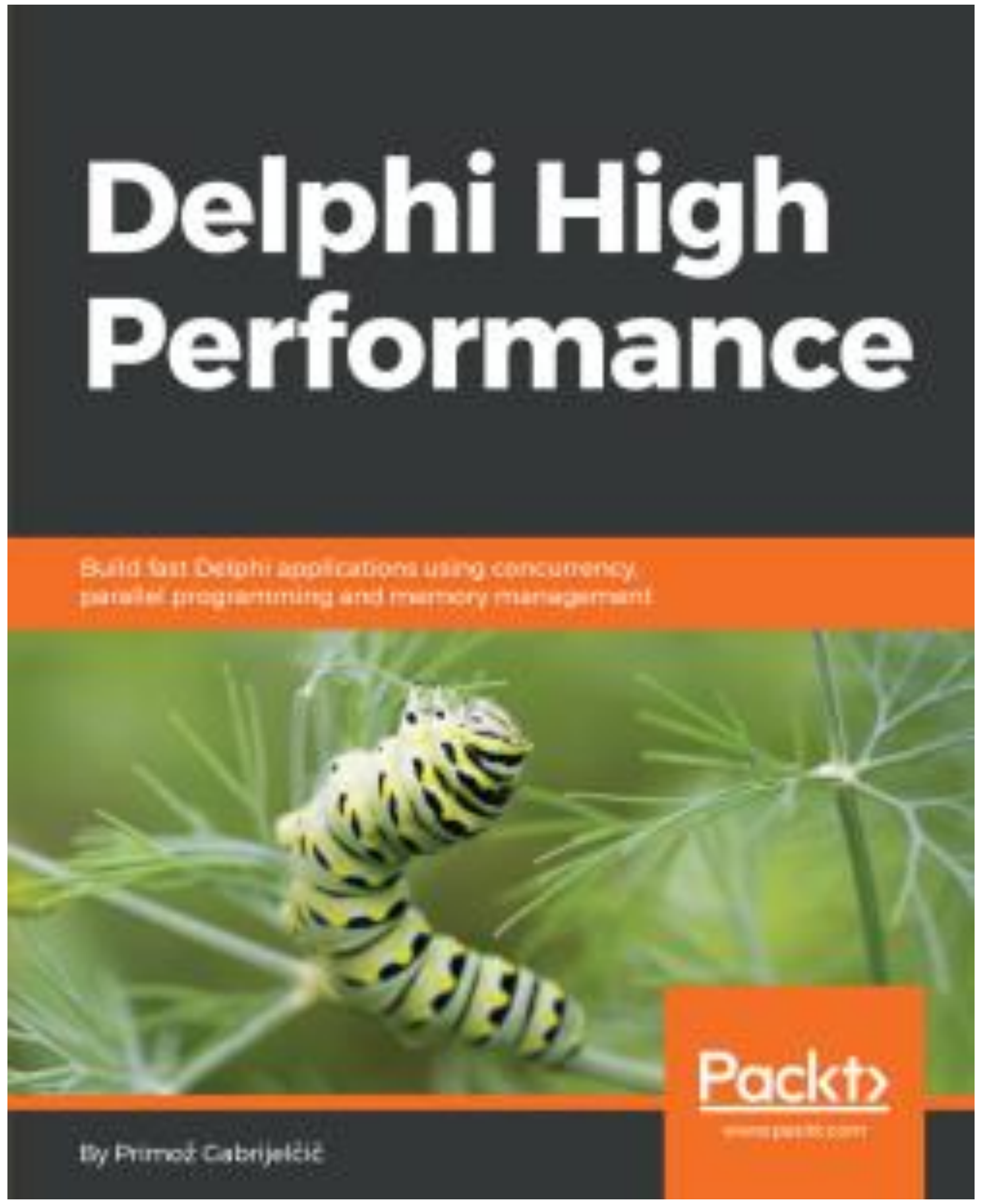
Parallel Programming

with

OmniThreadLibrary

Primož Gabrijelčič

thedelphigeek.org



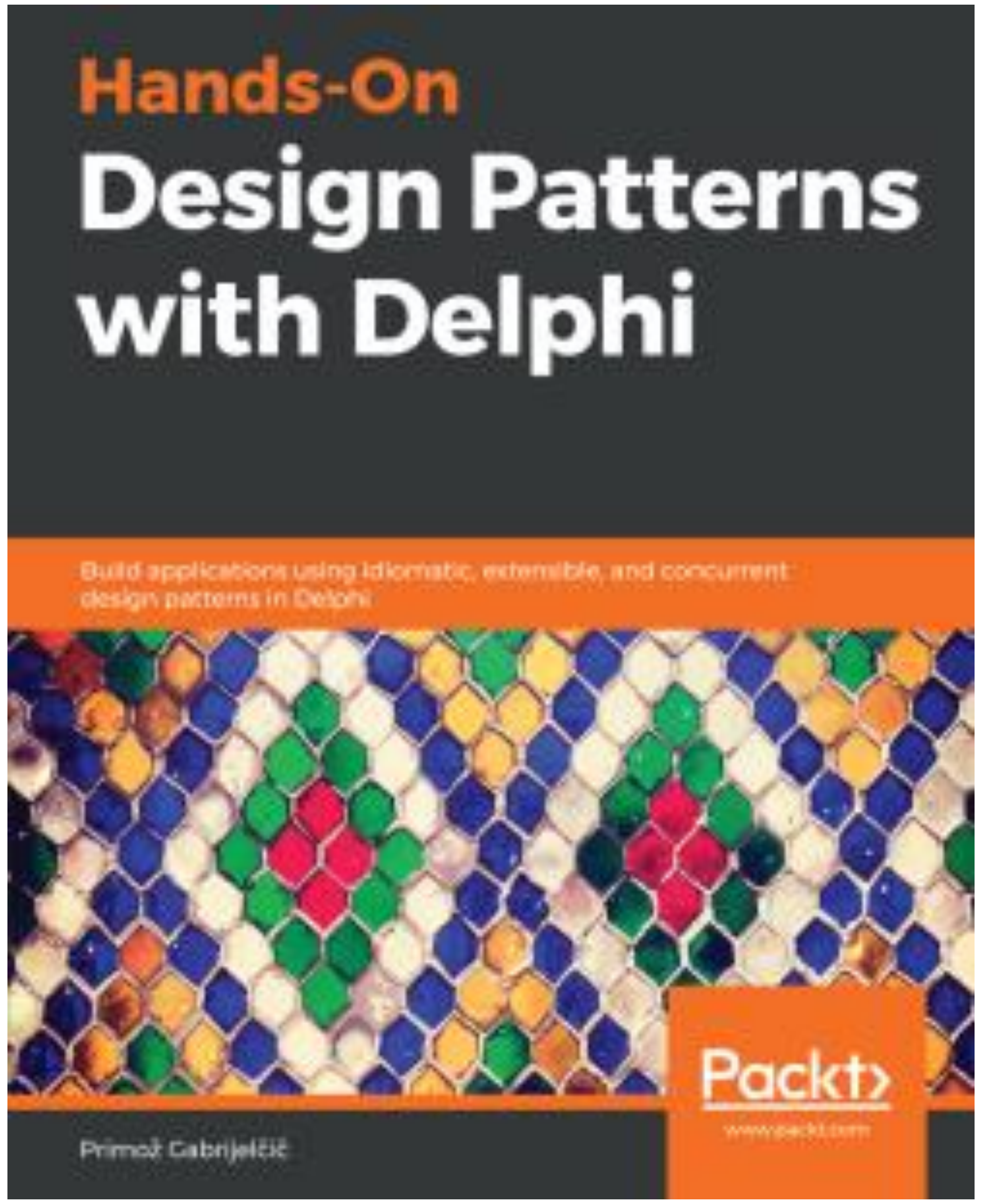
Delphi High Performance

Build fast Delphi applications using concurrency, parallel programming and memory management

Packt›

www.packt.com

By Primož Gabrijelčič



Hands-On

Design Patterns with Delphi

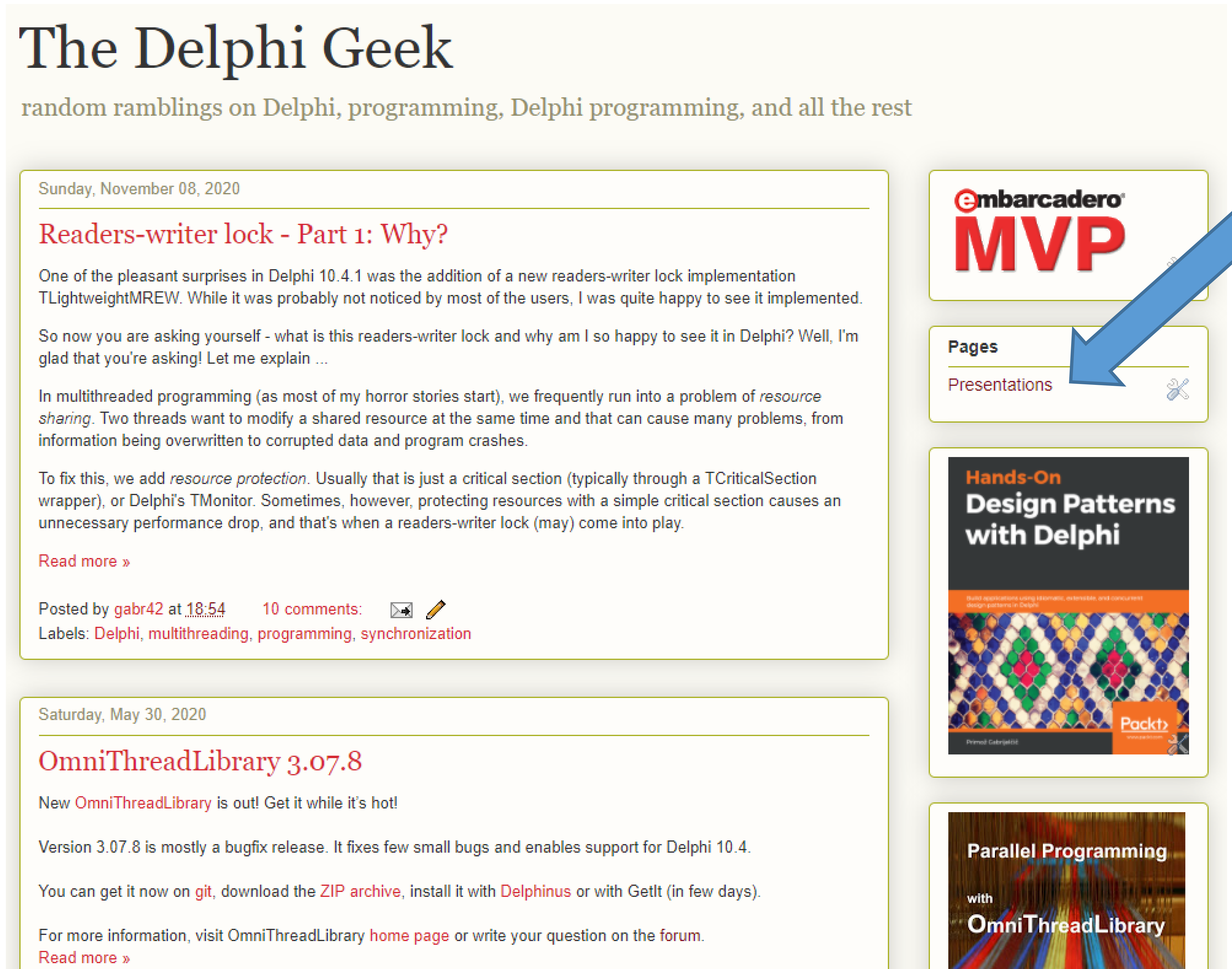
Build applications using idiomatic, extensible, and concurrent design patterns in Delphi

Packt›

www.packt.com

Primož Gabrijelčič

www.thedelphigeek.com



The Delphi Geek

random ramblings on Delphi, programming, Delphi programming, and all the rest

Sunday, November 08, 2020

Readers-writer lock - Part 1: Why?

One of the pleasant surprises in Delphi 10.4.1 was the addition of a new readers-writer lock implementation TLightweightMREW. While it was probably not noticed by most of the users, I was quite happy to see it implemented.

So now you are asking yourself - what is this readers-writer lock and why am I so happy to see it in Delphi? Well, I'm glad that you're asking! Let me explain ...

In multithreaded programming (as most of my horror stories start), we frequently run into a problem of *resource sharing*. Two threads want to modify a shared resource at the same time and that can cause many problems, from information being overwritten to corrupted data and program crashes.

To fix this, we add *resource protection*. Usually that is just a critical section (typically through a TCriticalSection wrapper), or Delphi's TMonitor. Sometimes, however, protecting resources with a simple critical section causes an unnecessary performance drop, and that's when a readers-writer lock (may) come into play.

Read more »

Posted by gabr42 at 18:54 10 comments:

Labels: Delphi, multithreading, programming, synchronization

Saturday, May 30, 2020

OmniThreadLibrary 3.07.8

New OmniThreadLibrary is out! Get it while it's hot!

Version 3.07.8 is mostly a bugfix release. It fixes few small bugs and enables support for Delphi 10.4.

You can get it now on git, download the ZIP archive, install it with Delphinus or with GetIt (in few days).

For more information, visit OmniThreadLibrary home page or write your question on the forum.

Read more »

embarcadero MVP
Pages
Presentations
Hands-On Design Patterns with Delphi
Packt
Parallel Programming with OmniThreadLibrary

Performance

Performance

- Running “fast enough”

- Raw speed
- Responsiveness
    - Non-blocking

Improving performance

- Find the problem – **Measure!**
- **Fix the algorithm**
- Fine tune the code
- Add parallelism
- Use external library
- Rewrite in assembler

Fixing the algorithm

- Find a better algorithm
- Run less code

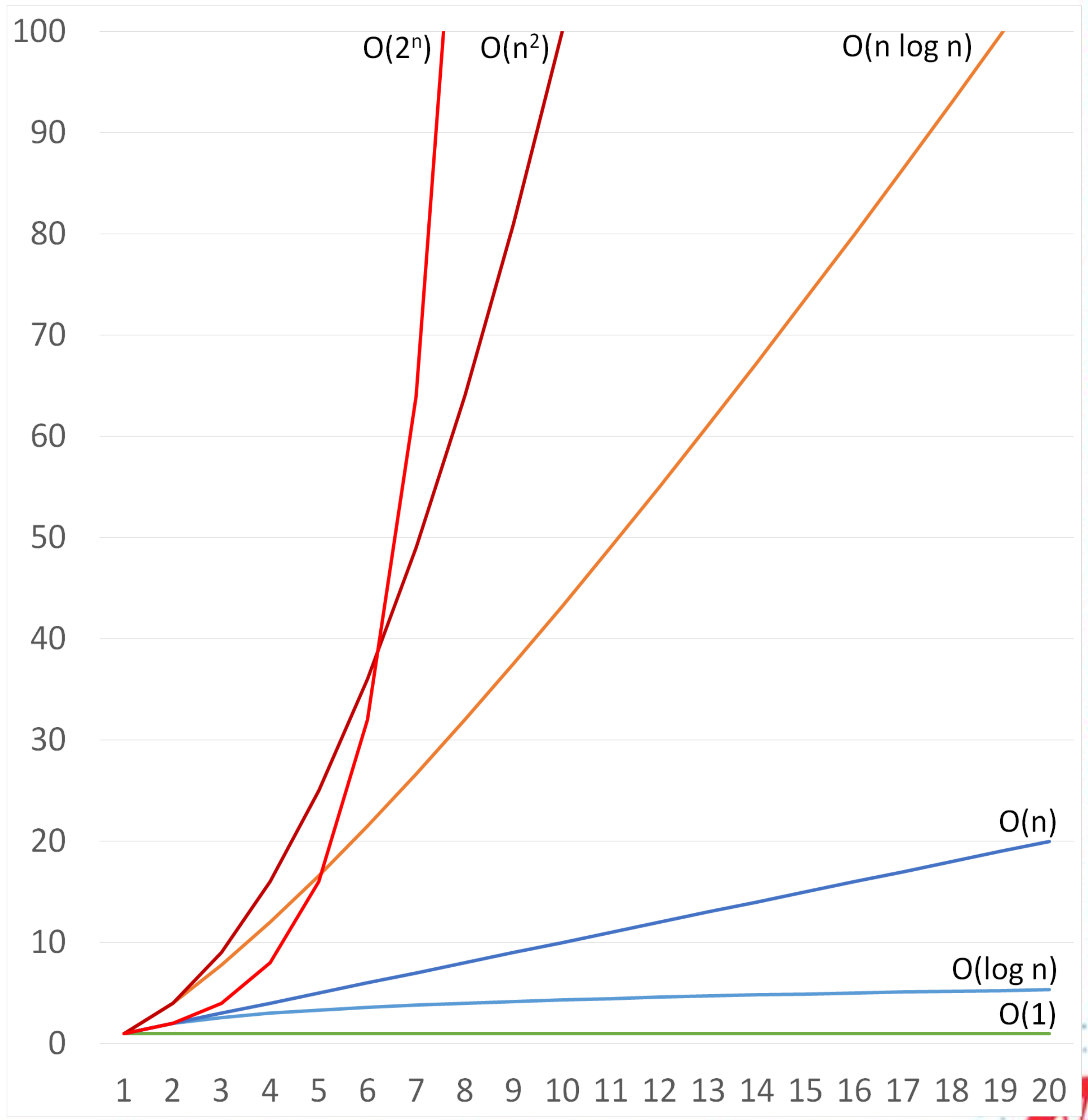
Choosing a better algorithm

Algorithm complexity

http://bigocheatsheet.com/

https://www.geeksforgeeks.org/data-structures/

- O() **Describes how algorithm slows down if data size is increased by a factor of *n***
- O(1) accessing array elements
- O(log n) searching in ordered list
- O(n) linear search
- O(n log n) quick sort (average)
- O(n^2) quick sort (worst), naive sort (bubblesort, insertion, selection)
- O(c^n) recursive Fibonacci, travelling salesman



Running less code

Running less code

- If a part of program is slow, don't execute it so much
- If a part of program is slow, don't execute it at all

“Don't execute it so much”

- Don't update UI thousands of times per second
- Call BeginUpdate/EndUpdate
- Don't send around millions of messages per second

“Don't execute it at all”

- UI virtualization
    - Virtual listbox
    - Virtual TreeView

- Memoization
    - Caching
    - Dynamic programming
    - TGpCache<K,V>
        - O(1) all operations
        - GpLists.pas, https://github.com/gabr42/GpDelphiUnits/

Learn more

Learn more

- Books
  - Julian Bucknall, **Algorithms and Data Structures**
  - Primož Gabrijelčič, **Delphi High Performance**
  - Robert Sedgewick & Kevin Wayne, **Algorithms**
- Web
  - Algorithms, 4th Edition, https://algs4.cs.princeton.edu/home/
  - Udacity, edX, Udemy, Coursera …
  - https://www.geeksforgeeks.org/data-structures/
  - https://www.geeksforgeeks.org/fundamentals-of-algorithms/
- Forums
  - Delphi-PRAXIS, www.delphipraxis.net, en.delphipraxis.net