A new datatype in LabVIEW

Sets and Maps

Peter Horn – CLA/CTA

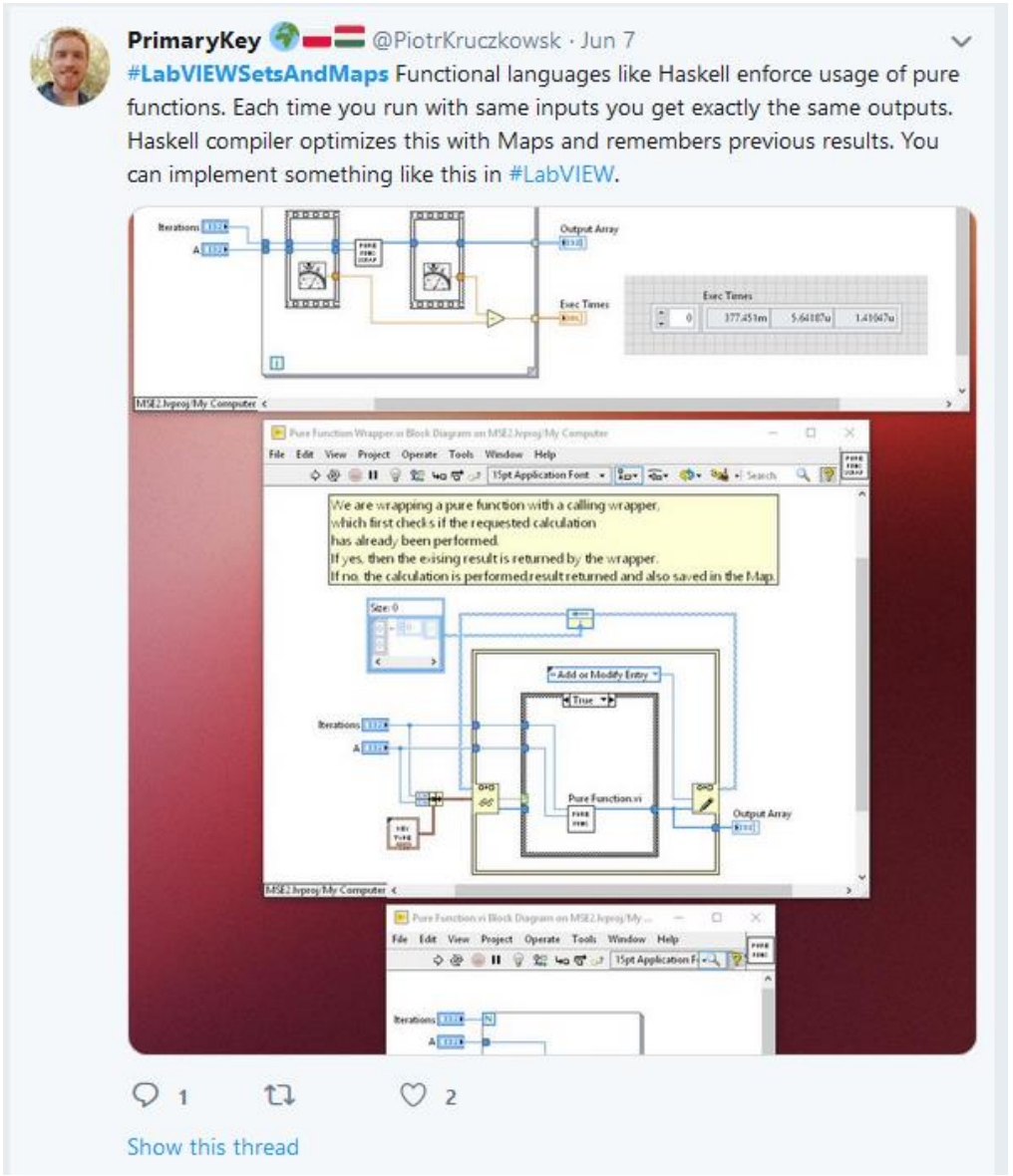
Senior Applications Engineer - NI

#OurGiantsAreFemale

- Margaret Hamilton
  - Wrote the code for the 1969 Apollo 11 mission to the moon
  - Included priority alarm displays to interrupt the astronauts normal displays in an emergency during landing – giving them a go/no-go decision
  - Developed early software for predicting weather on LGP-30 and PDP-1 computers
  - Contributed to publications on chaos theory

NATIONAL INSTRUMENTS

Thanks to PrimaryKey (Piotr Kruczkowsk)



Thanks also Christian Altenbach

- https://www.youtube.com/watch?v=5L9tqv3a5TU&feature=youtu.be
- https://labviewwiki.org/wiki/NIWeek_2019

- **From Variant Attributes to Sets and Maps (New in 2019)** by Christian Altenbach, Research Ophthalmologist, UCLA (slides)

LabVIEW 2019 introduces two new data types: sets and maps. You can use these to achieve simpler, more intuitive solutions to common programming tasks and replace existing code based on variant attributes. Learn how to adapt older code and explore innovative examples where these new data types really shine.

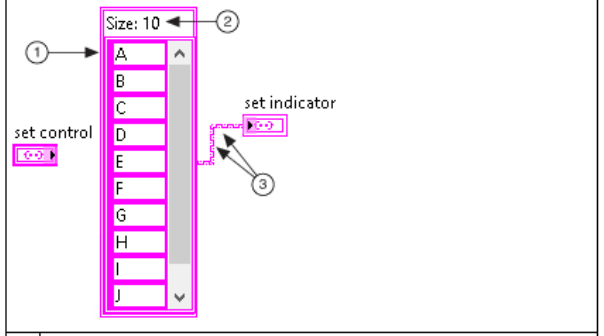
NATIONAL INSTRUMENTS

Agenda

- What are LabVIEW Sets and Maps?
- What are they good for, and why are they brilliant?
- Example use cases
  - Sets
  - Maps
- Some other cool stuff built in to the environment
- Performance comparison

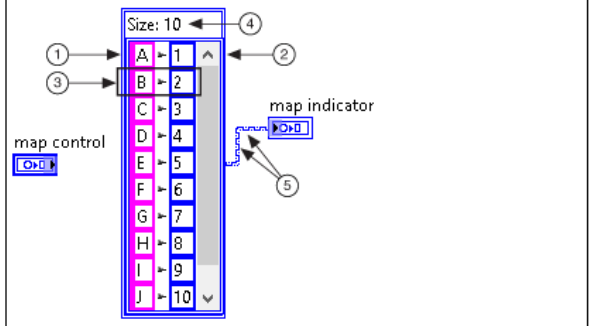
What are LabVIEW Sets and Maps?

- **Set**—A collection of unique elements of the same data type.

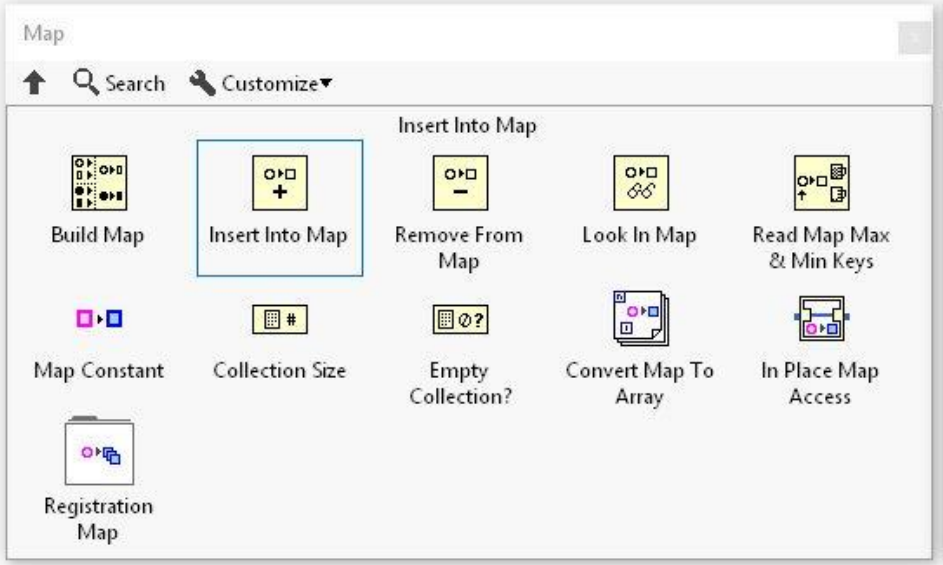


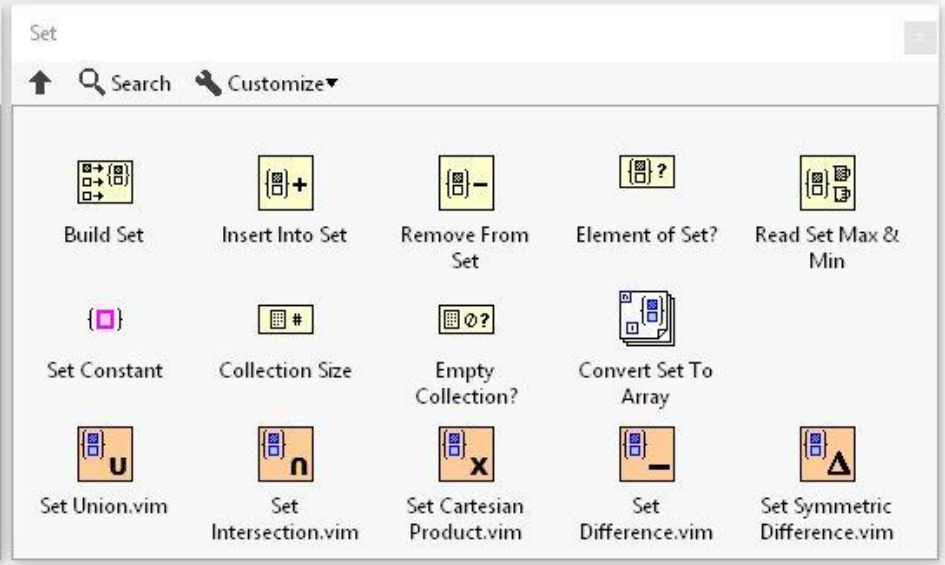
1	Element
2	Size—Number of elements in the set
3	Wires of set—Wire color matches the data type color of the element

- **Map**—A collection of entries consisting of key-value pairs where all the keys are unique. The key and value can each be any data type. Maps are also known as dictionaries because the key is used to look up a value.



1	Key
2	Value
3	Entry—A pair of associated key and value
4	Size—Number of entries in the map
5	Wires of map—Wire color matches the data type color of the value





NATIONAL INSTRUMENTS

What are they good for, and why are they brilliant?

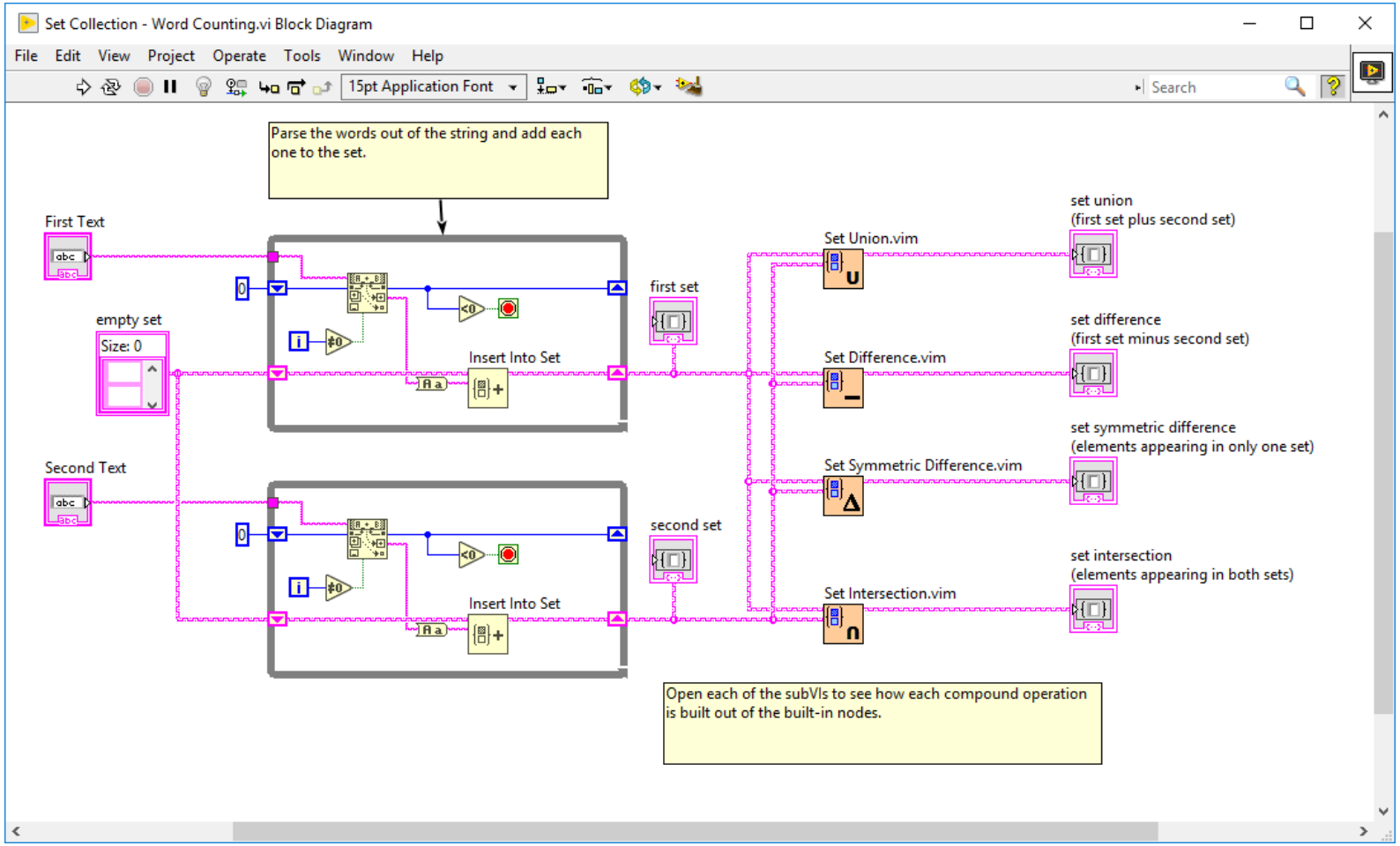
- Having the information is useless unless we can find it efficiently

- Sets and Maps are optimised for storing information and efficient searching
  - Does this entry exist? (sets: **element**, maps: **key**)
  - If it exists, what is the value (maps: **value**)

- Sets and Maps are optimised for dynamic changes to the data
  - Add entry (maps, sets)
  - Remove entry (maps, sets)
  - Replace value (IPE (3 modes), maps)

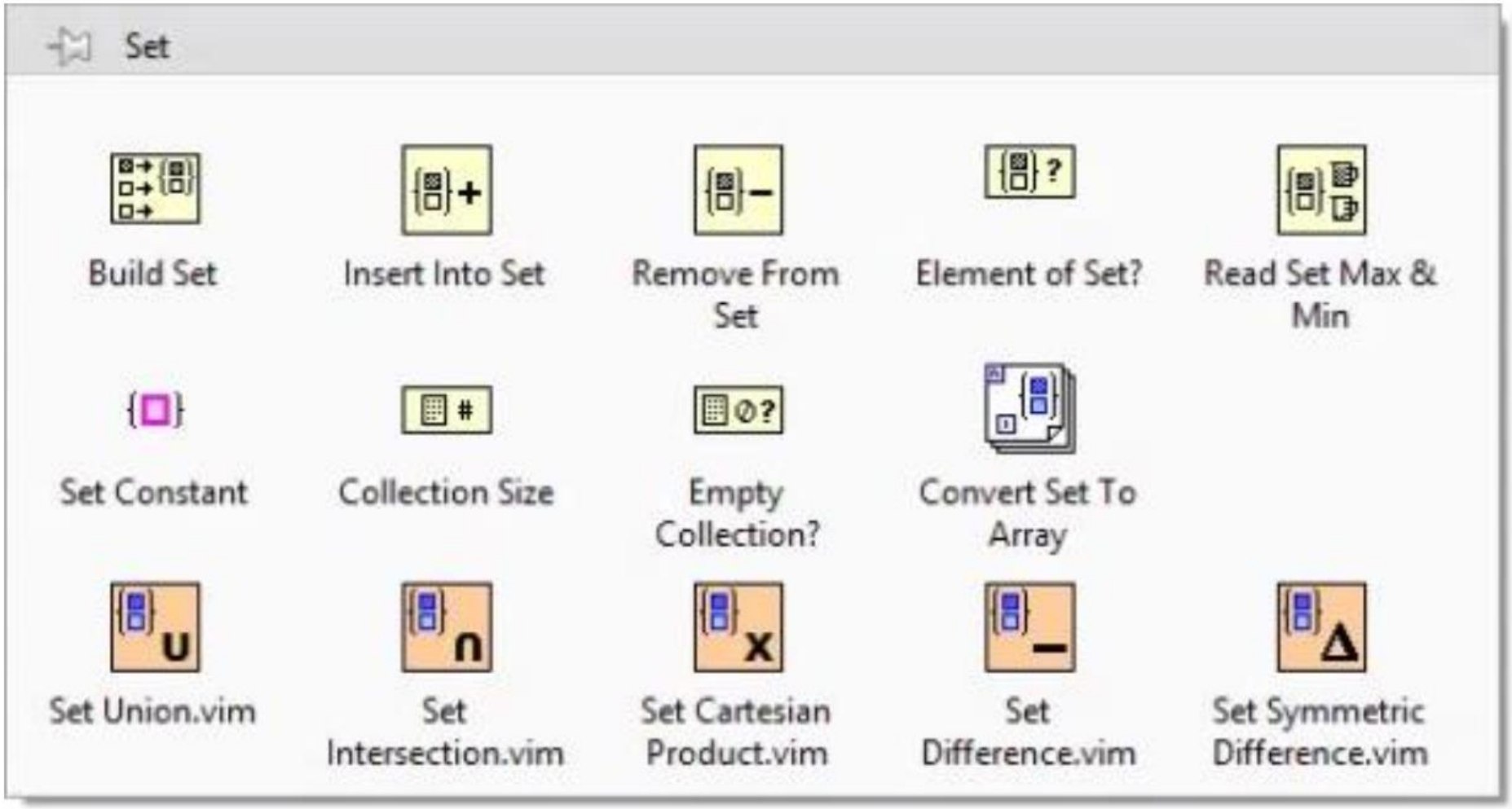
Sets

- Collection of **unique** elements **sorted** into order
  - As they are sorted, it is very quick to check whether a given element is present or to iterate through the collection

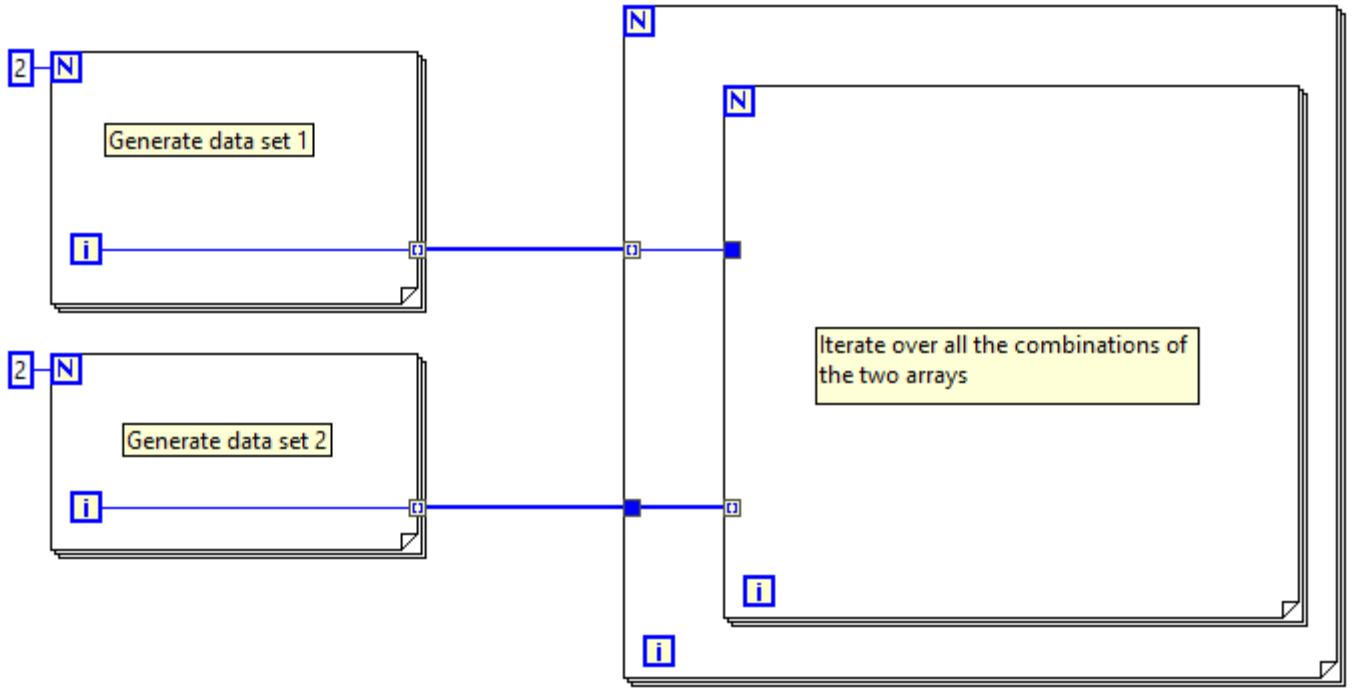


NATIONAL INSTRUMENTS

Sets

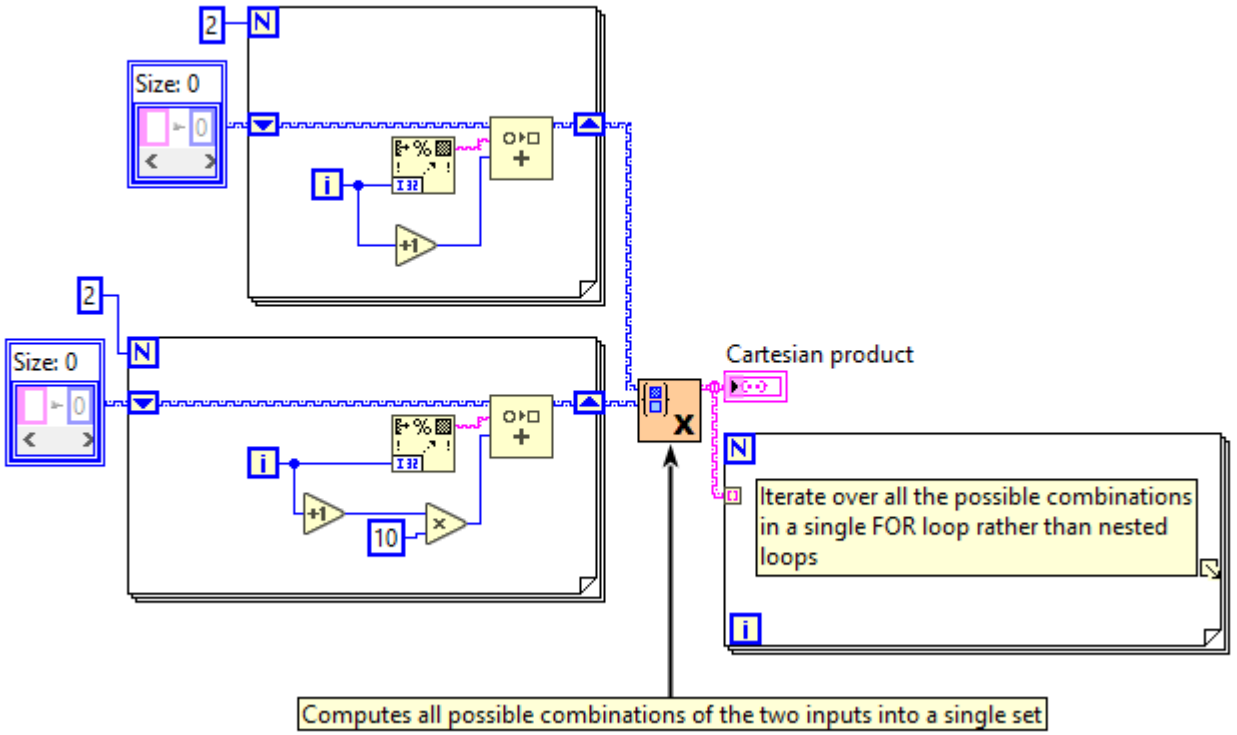

Set
Build Set
Insert Into Set
Remove From Set
Element of Set?
Read Set Max & Min
Set Constant
Collection Size
Empty Collection?
Convert Set To Array
Set Union.vim
Set Intersection.vim
Set Cartesian Product.vim
Set Difference.vim
Set Symmetric Difference.vim

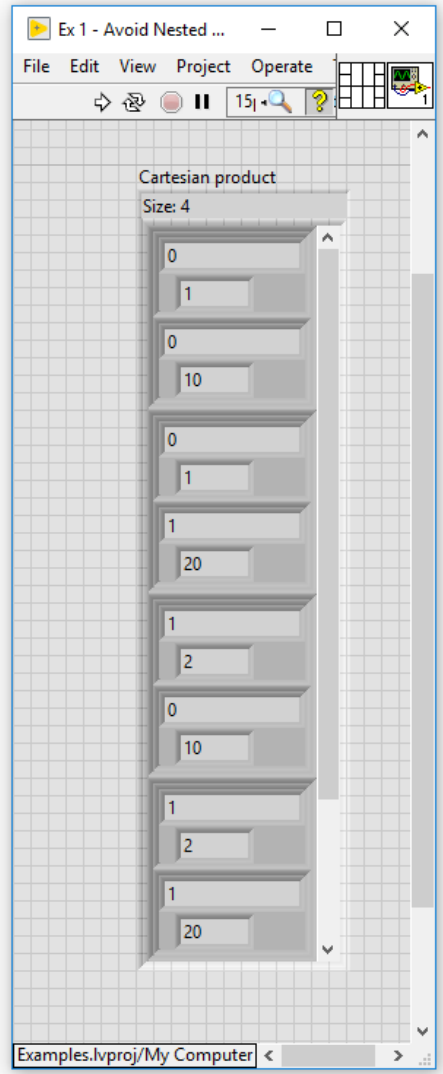
Avoid nested loops



NATIONAL INSTRUMENTS

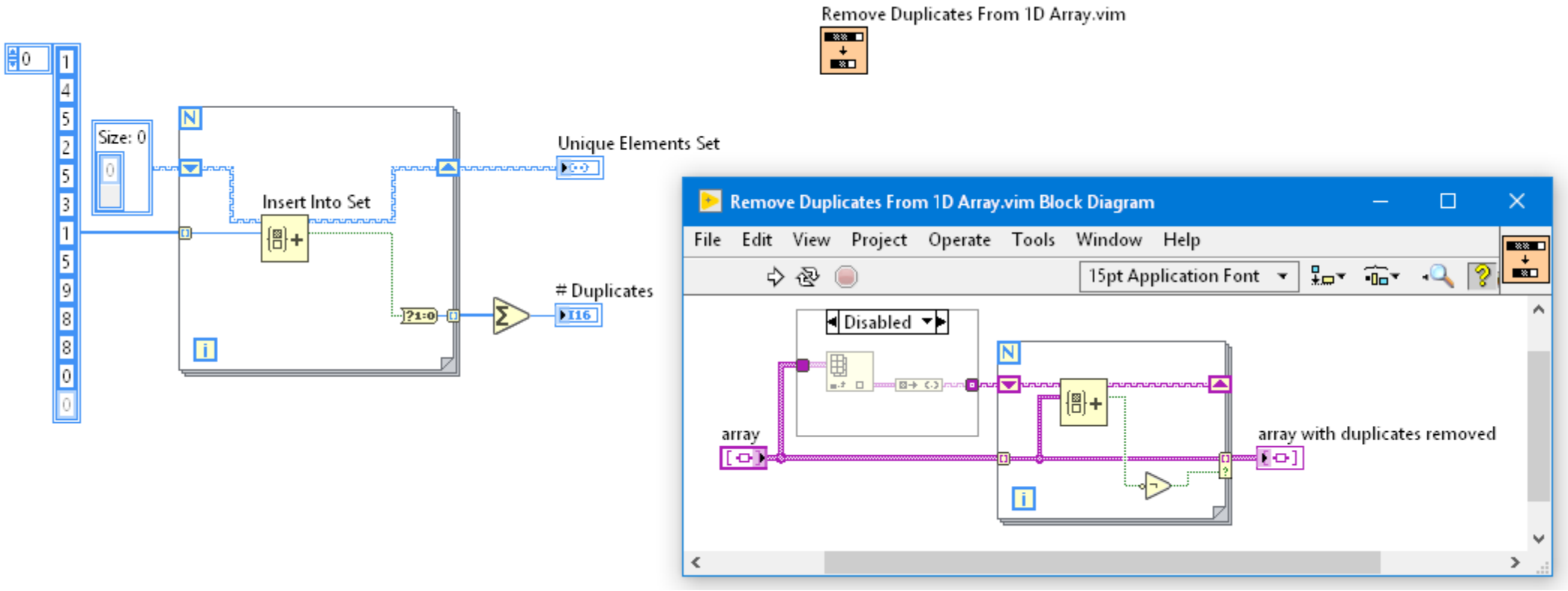
Avoid nested loops





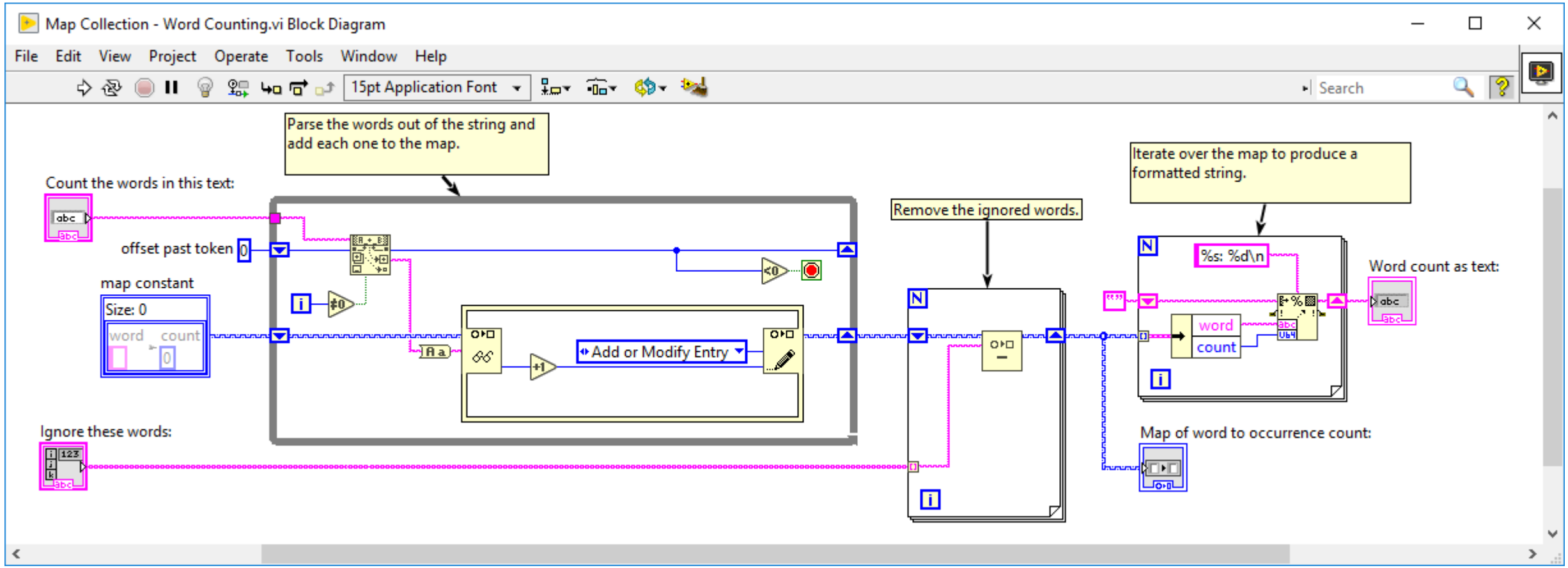
NATIONAL INSTRUMENTS

Finding unique elements in an array



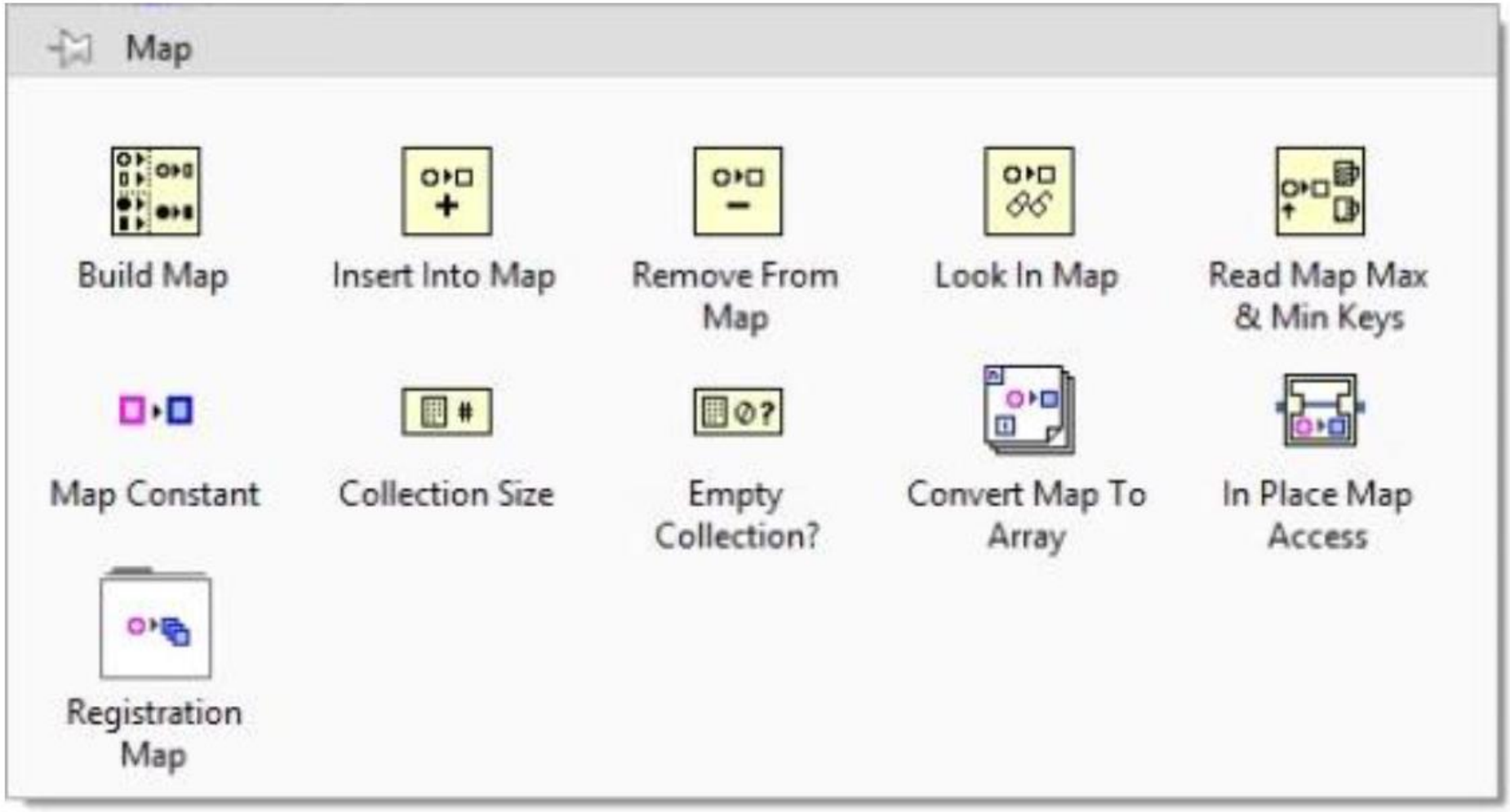
Maps

- Collection of **unique** keys with values. Both keys and values can have custom data types



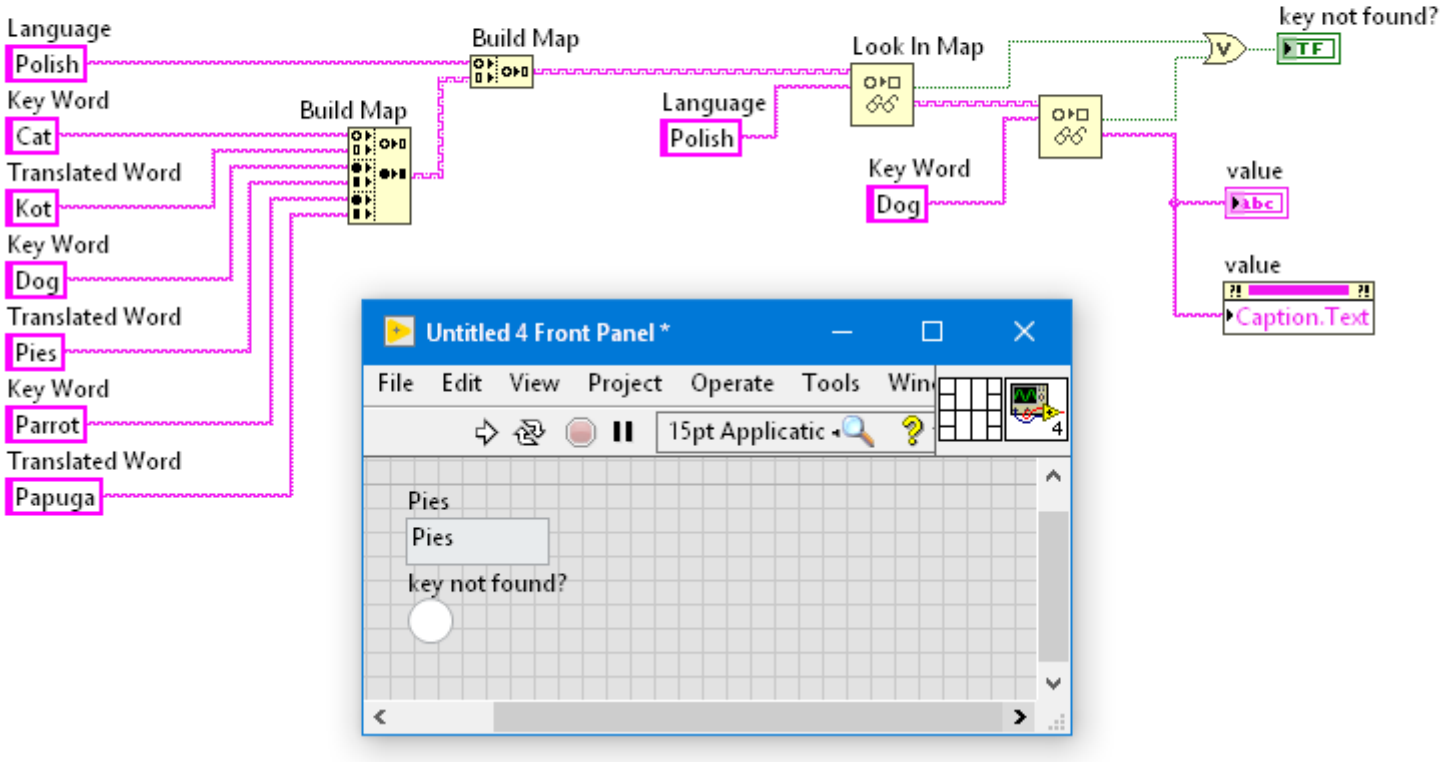
NATIONAL INSTRUMENTS

Maps

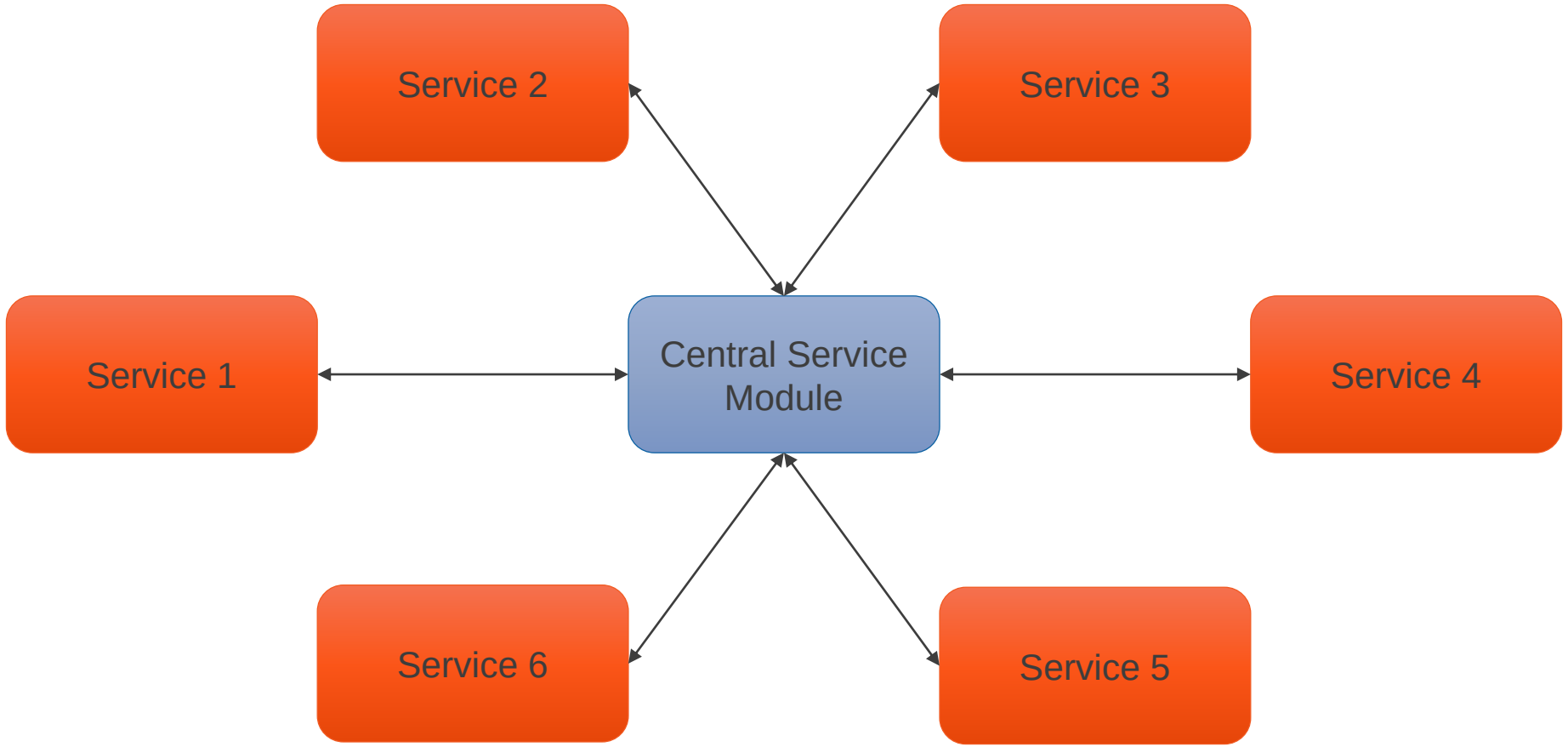

Map
Build Map
Insert Into Map
Remove From Map
Look In Map
Read Map Max & Min Keys
Map Constant
Collection Size
Empty Collection?
Convert Map To Array
In Place Map Access
Registration Map

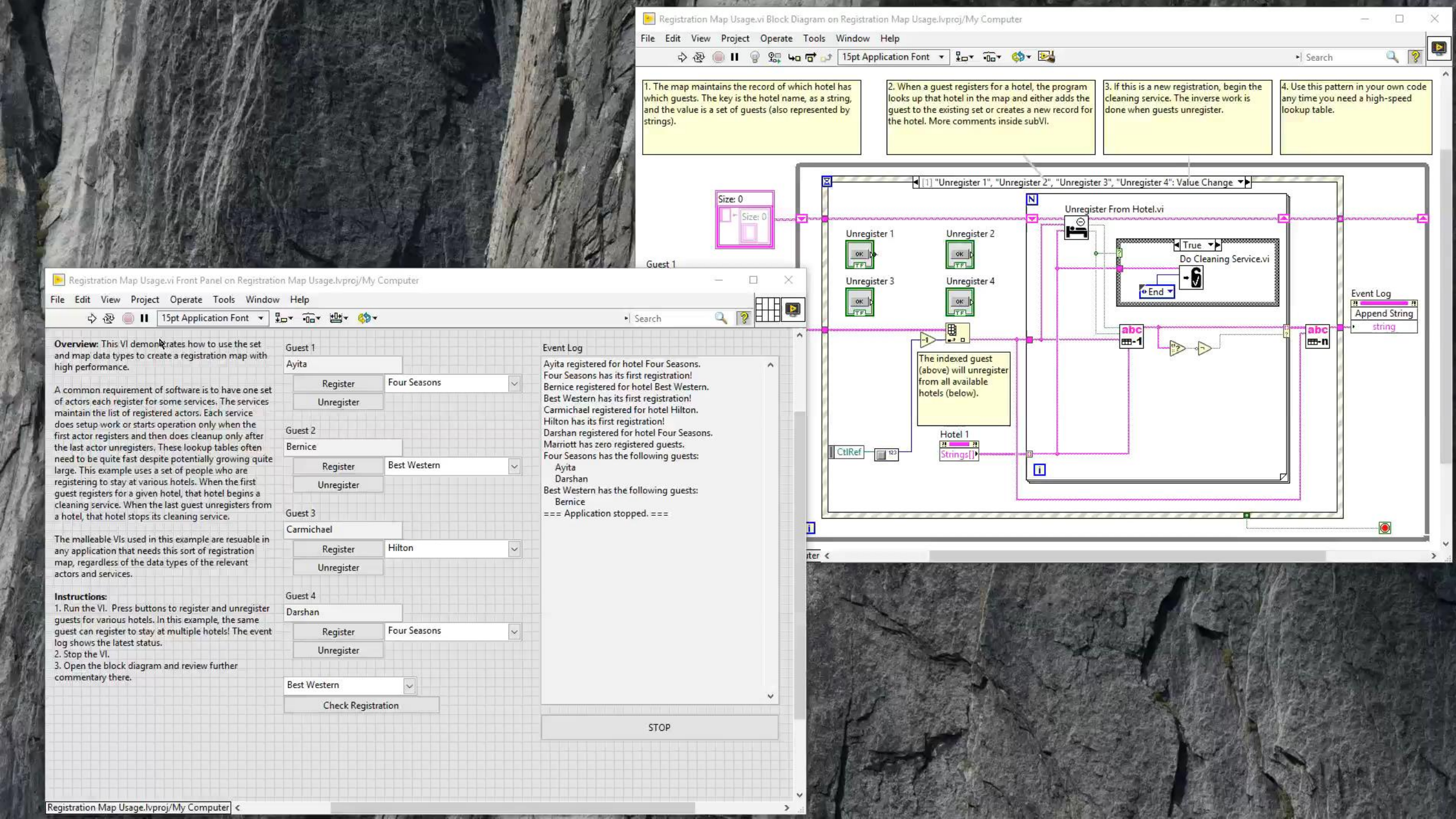
Build a translation utility

Map of Maps represents multiple languages with multiple words in each language. Simply look-up the language and the word.

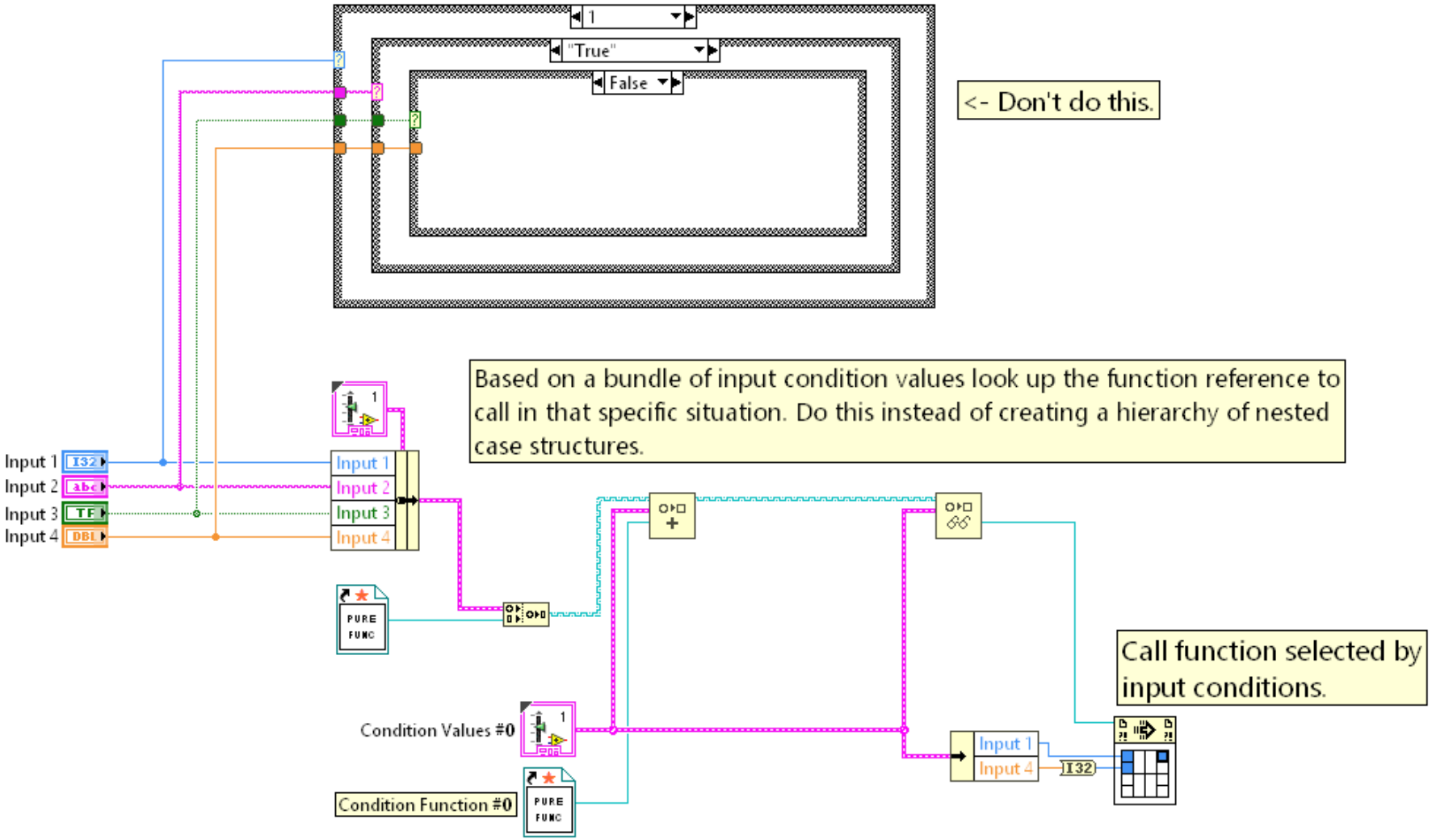


Service management



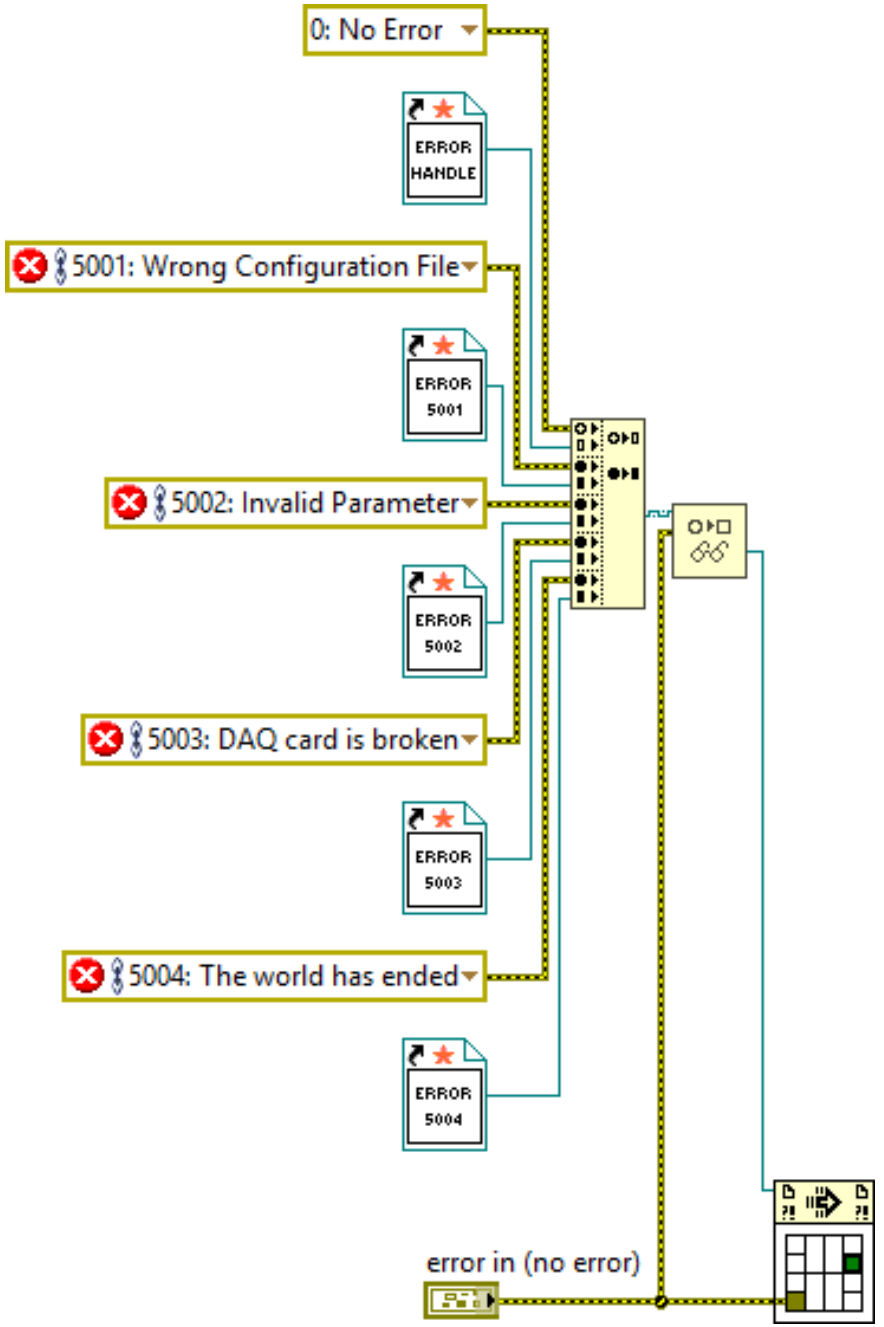

Registration Map Usage.vi Block Diagram on Registration Map Usage.lvproj/My Computer
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
1. The map maintains the record of which hotel has which guests. The key is the hotel name, as a string, and the value is a set of guests (also represented by strings).
2. When a guest registers for a hotel, the program looks up that hotel in the map and either adds the guest to the existing set or creates a new record for the hotel. More comments inside subVI.
3. If this is a new registration, begin the cleaning service. The inverse work is done when guests unregister.
4. Use this pattern in your own code any time you need a high-speed lookup table.
Size: 0
Size: 0
Guest 1
[1] "Unregister 1", "Unregister 2", "Unregister 3", "Unregister 4": Value Change
Unregister From Hotel.vi
Unregister 1
Unregister 2
Unregister 3
Unregister 4
True
Do Cleaning Service.vi
End
-6
Event Log
Append String
string
abc
-1
abc
-n
The indexed guest (above) will unregister from all available hotels (below).
CtlRef
Hotel 1
Strings[]
Registration Map Usage.vi Front Panel on Registration Map Usage.lvproj/My Computer
File Edit View Project Operate Tools Window Help
15pt Application Font
Search
Overview: This VI demonstrates how to use the set and map data types to create a registration map with high performance.
A common requirement of software is to have one set of actors each register for some services. The services maintain the list of registered actors. Each service does setup work or starts operation only when the first actor registers and then does cleanup only after the last actor unregisters. These lookup tables often need to be quite fast despite potentially growing quite large. This example uses a set of people who are registering to stay at various hotels. When the first guest registers for a given hotel, that hotel begins a cleaning service. When the last guest unregisters from a hotel, that hotel stops its cleaning service.
The malleable VIs used in this example are resuable in any application that needs this sort of registration map, regardless of the data types of the relevant actors and services.
Instructions:
1. Run the VI. Press buttons to register and unregister guests for various hotels. In this example, the same guest can register to stay at multiple hotels! The event log shows the latest status.
2. Stop the VI.
3. Open the block diagram and review further commentary there.
Guest 1
Ayita
Register
Four Seasons
Unregister
Guest 2
Bernice
Register
Best Western
Unregister
Guest 3
Carmichael
Register
Hilton
Unregister
Guest 4
Darshan
Register
Four Seasons
Unregister
Best Western
Check Registration
Event Log
Ayita registered for hotel Four Seasons.
Four Seasons has its first registration!
Bernice registered for hotel Best Western.
Best Western has its first registration!
Carmichael registered for hotel Hilton.
Hilton has its first registration!
Darshan registered for hotel Four Seasons.
Marriott has zero registered guests.
Four Seasons has the following guests:
Ayita
Darshan
Best Western has the following guests:
Bernice
=== Application stopped. ===
STOP
Registration Map Usage.lvproj/My Computer

Decision trees



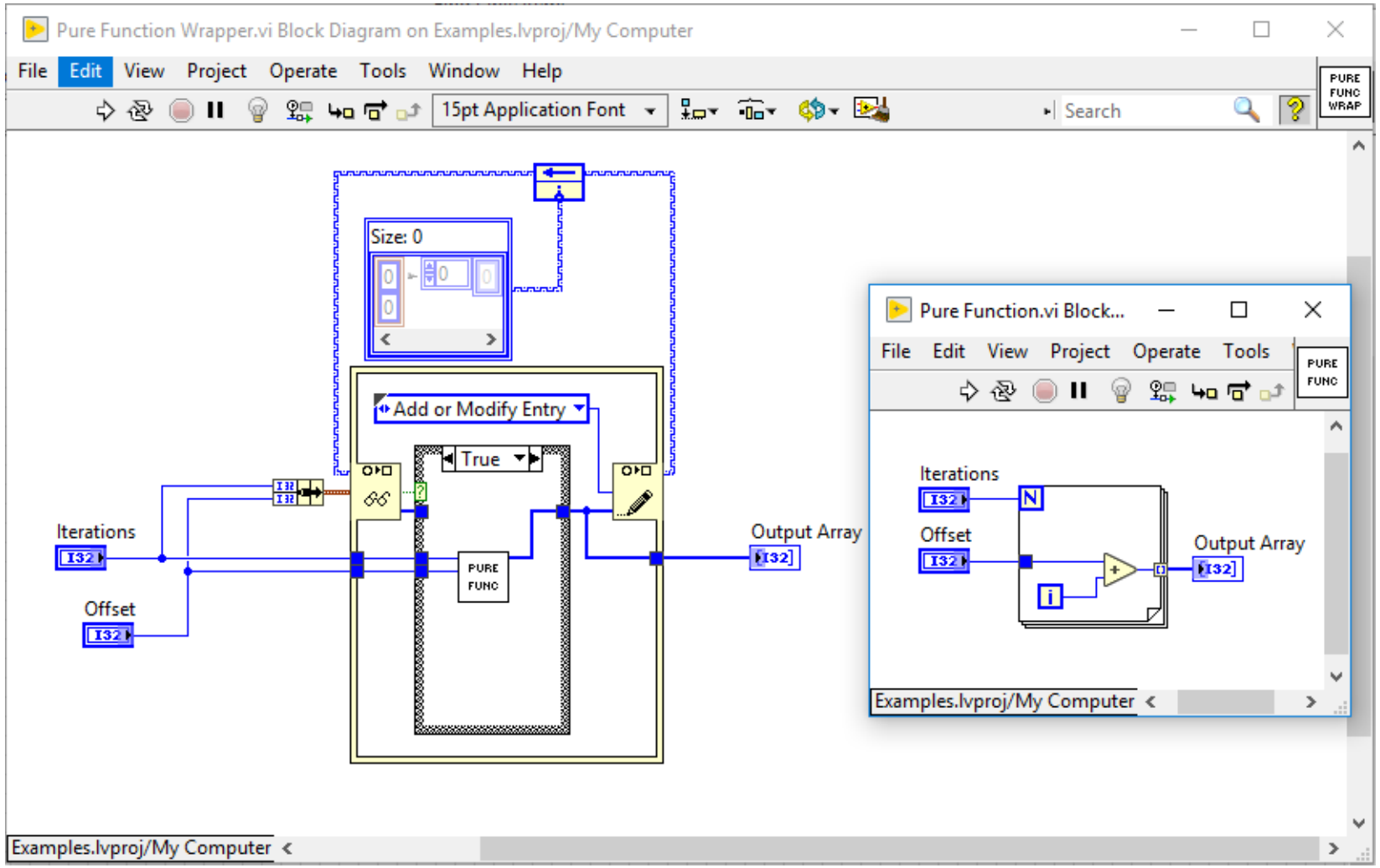
NATIONAL INSTRUMENTS

Decision trees



NATIONAL INSTRUMENTS

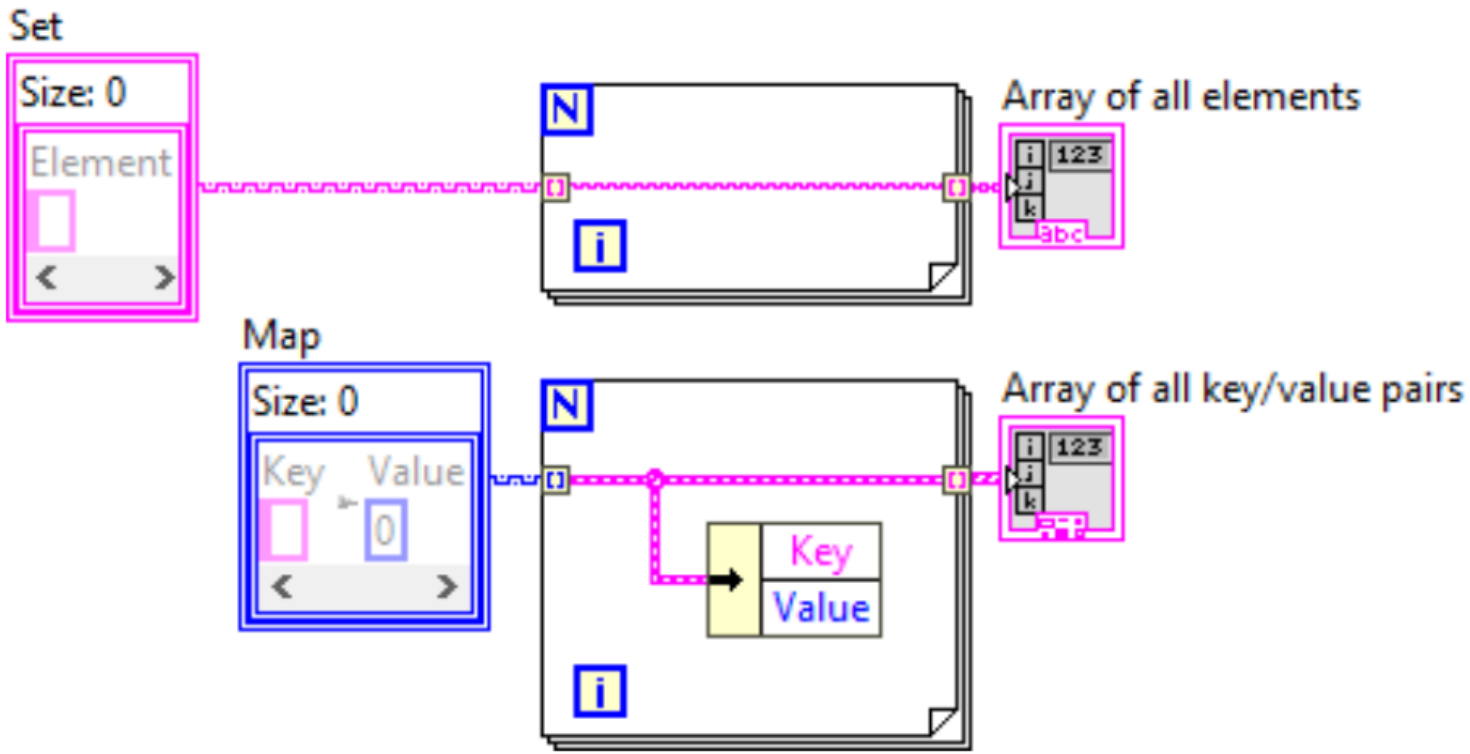
Remembering previous results



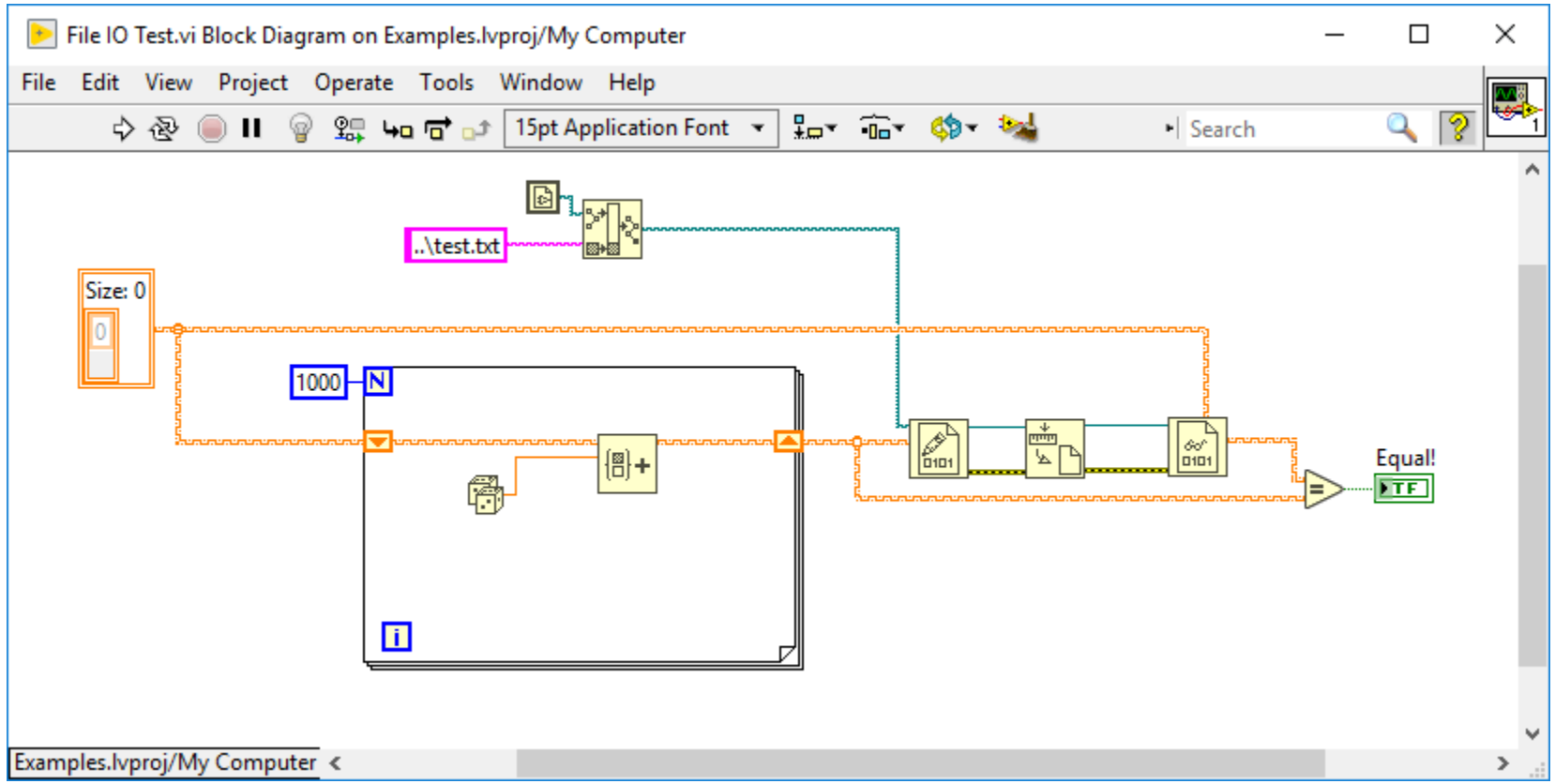
NATIONAL INSTRUMENTS

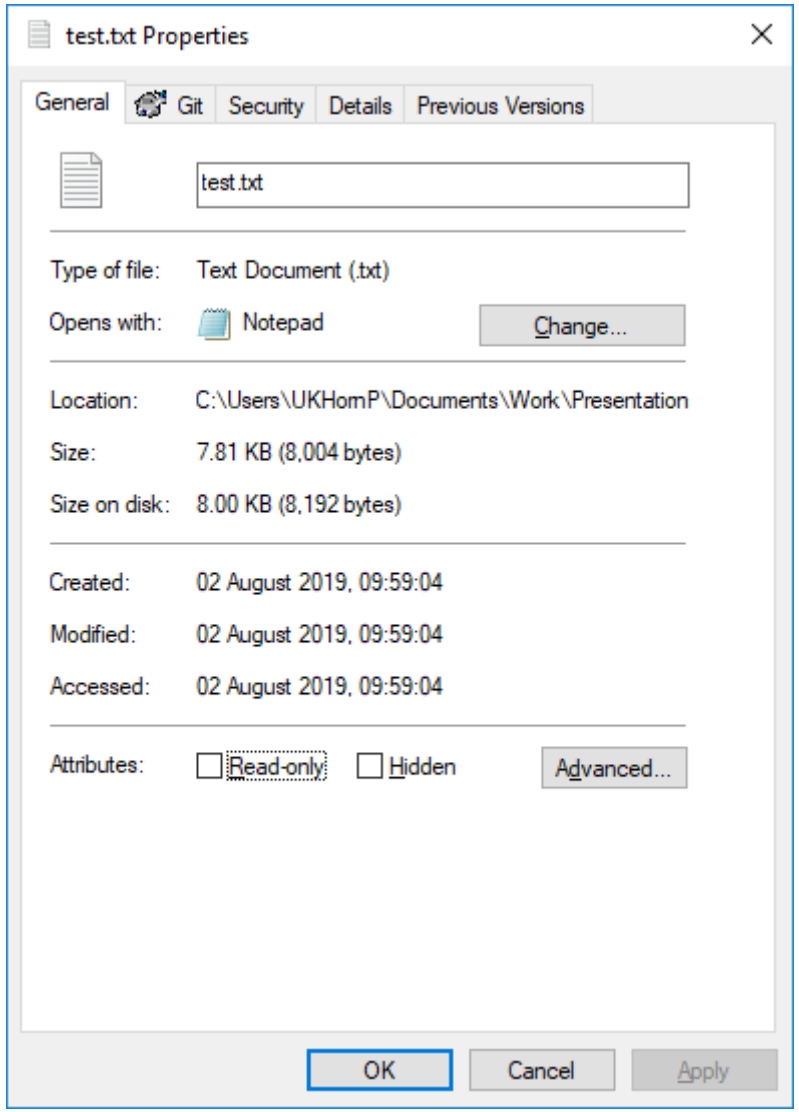
Indexing a FOR loop

- Set: autoindexing gives all elements sorted
- Map: autoindexing gives cluster of key/value pairs sorted by key
  - Good idea to name the key/value constants in the data type for self documenting code



File IO – Zero overhead





NATIONAL INSTRUMENTS

Performance

Re: LV2019 Maps vs. Variant Attributes: Performance?

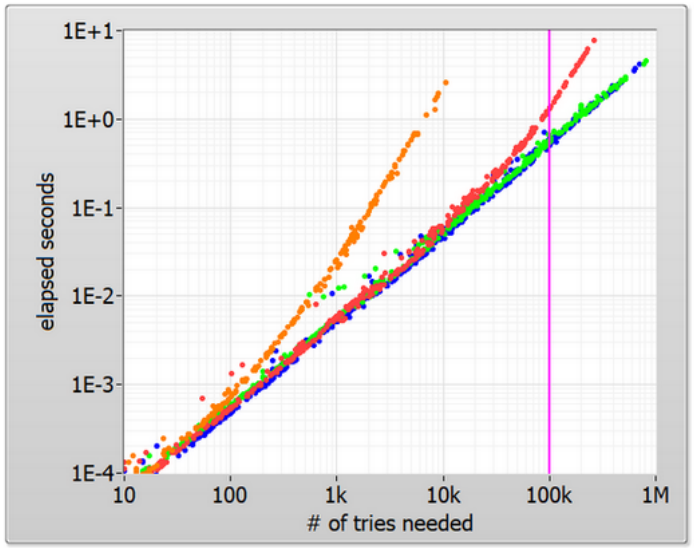
altenbach KNIGHT OF NI

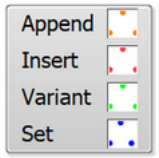
06-06-2019 01:39 PM - edited 06-06-2019 02:04 PM Options

> @altenbach wrote:
> And the map code is now consistently faster by a few % ...

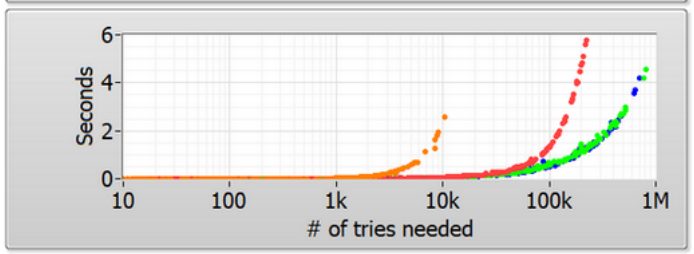
Here is a typical benchmark result. Note that Set results (blue) form the lower boundary at all x-values, but variant is about the same (within noise).

The code generates random lotto numbers and returns the # of tries and the time needed to generate lotto numbers that have already been generated during the same run. The collection of numbers to be compared grows linearly, so the code returns surprisingly fast.. The table shows the time and slope at 100k, i.e. where the cursor is. The slope for each curve is calculated from a quadratic polynomial fit (least abs. residual to ignore outliers) to the log-log data.





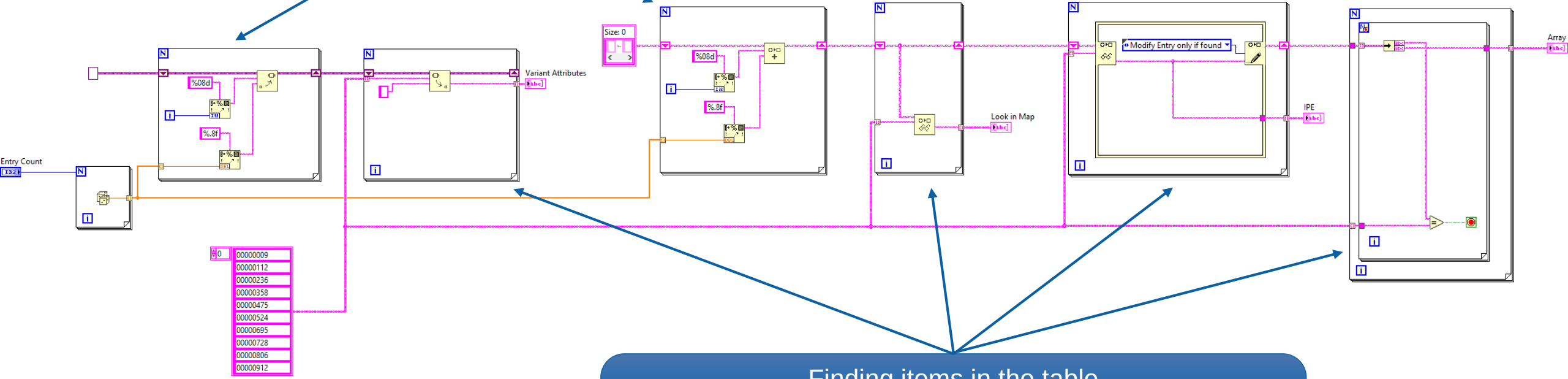
Slope	Time	x slower
2.74	850.61s	1596.50
1.34	1.24s	2.32
1.02	552.77ms	1.04
1.03	532.79ms	1.00



Note that if we divide the elapsed time by the # of tries, sets and variants are basically flat (slope~0), i.e. almost no penalty with collection size.

Performance – Maps vs variant attributes

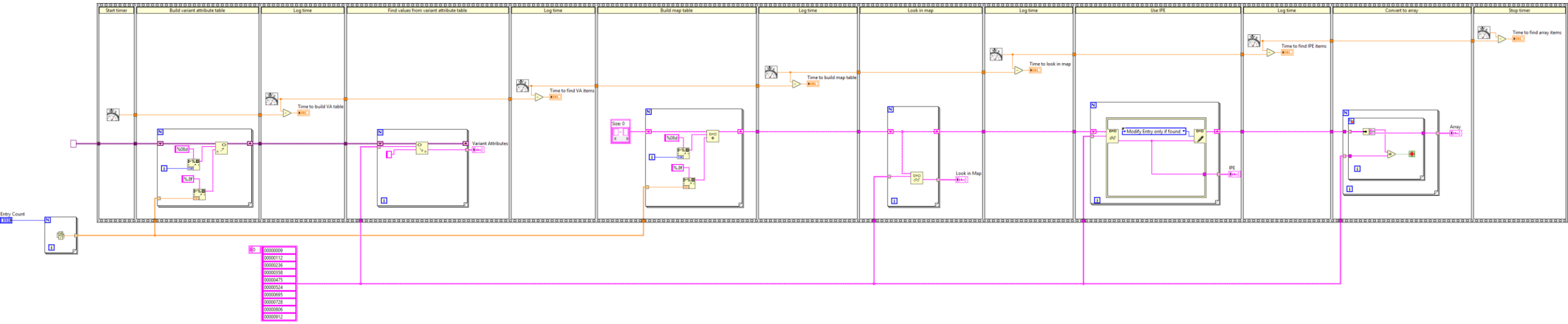
Building the table
Which will be fastest?
Variant attributes or maps



Finding items in the table
Which will be fastest?
Variant attributes, looking in maps, using the in place element structure or converting to an array

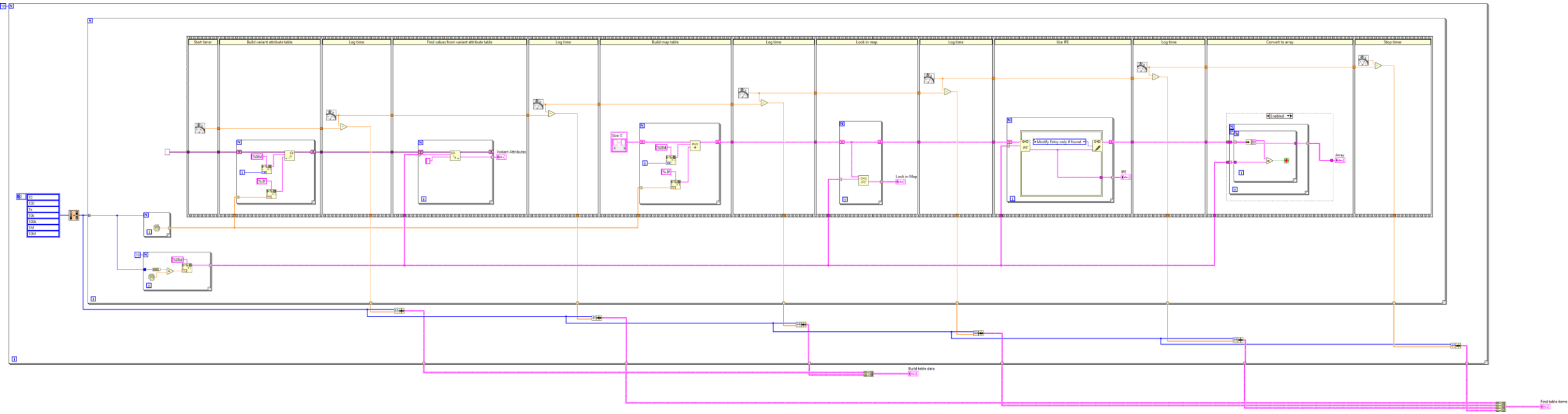
NATIONAL INSTRUMENTS

Performance – Maps vs variant attributes



NATIONAL INSTRUMENTS

Performance – Maps vs variant attributes

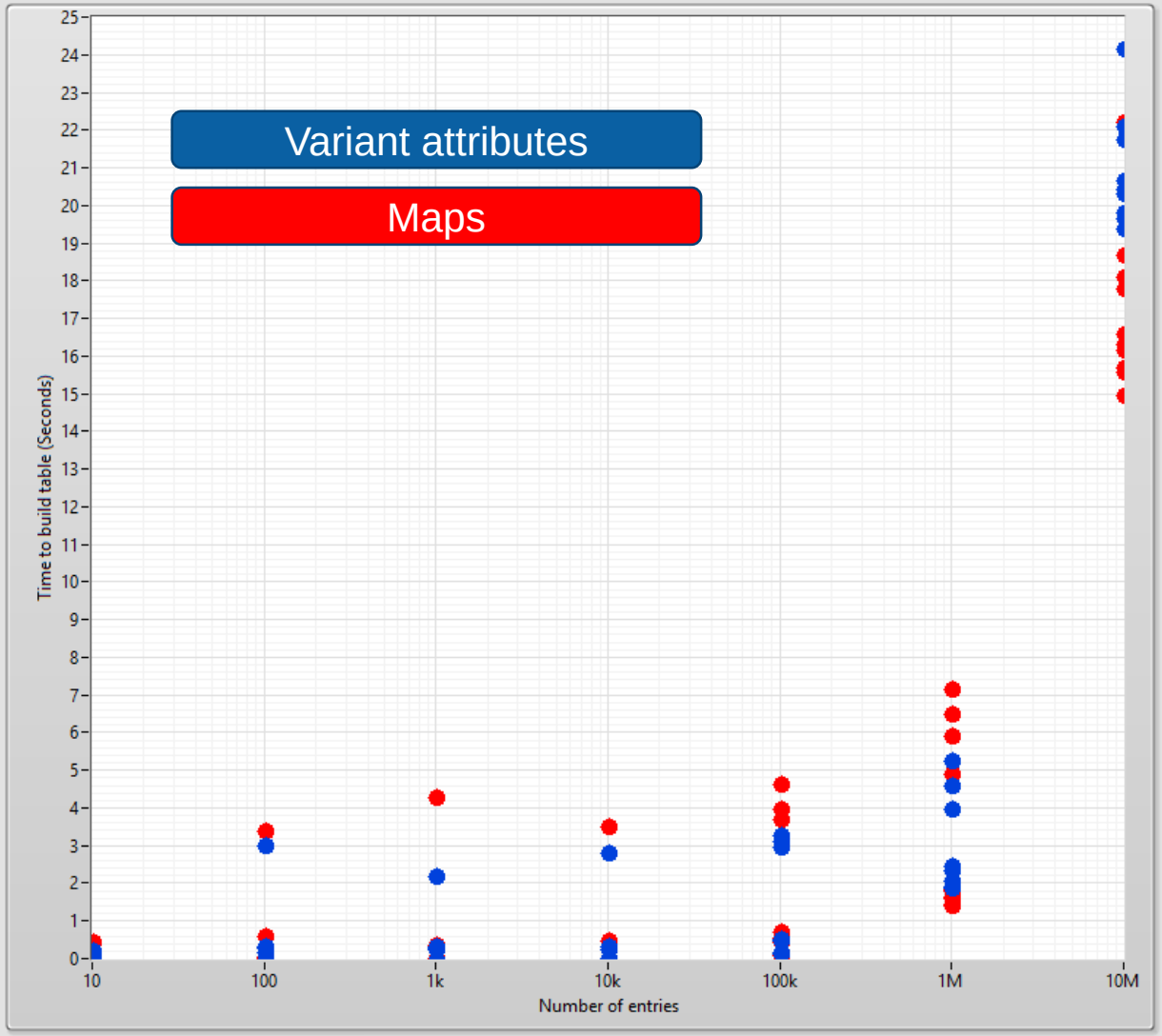


NATIONAL INSTRUMENTS

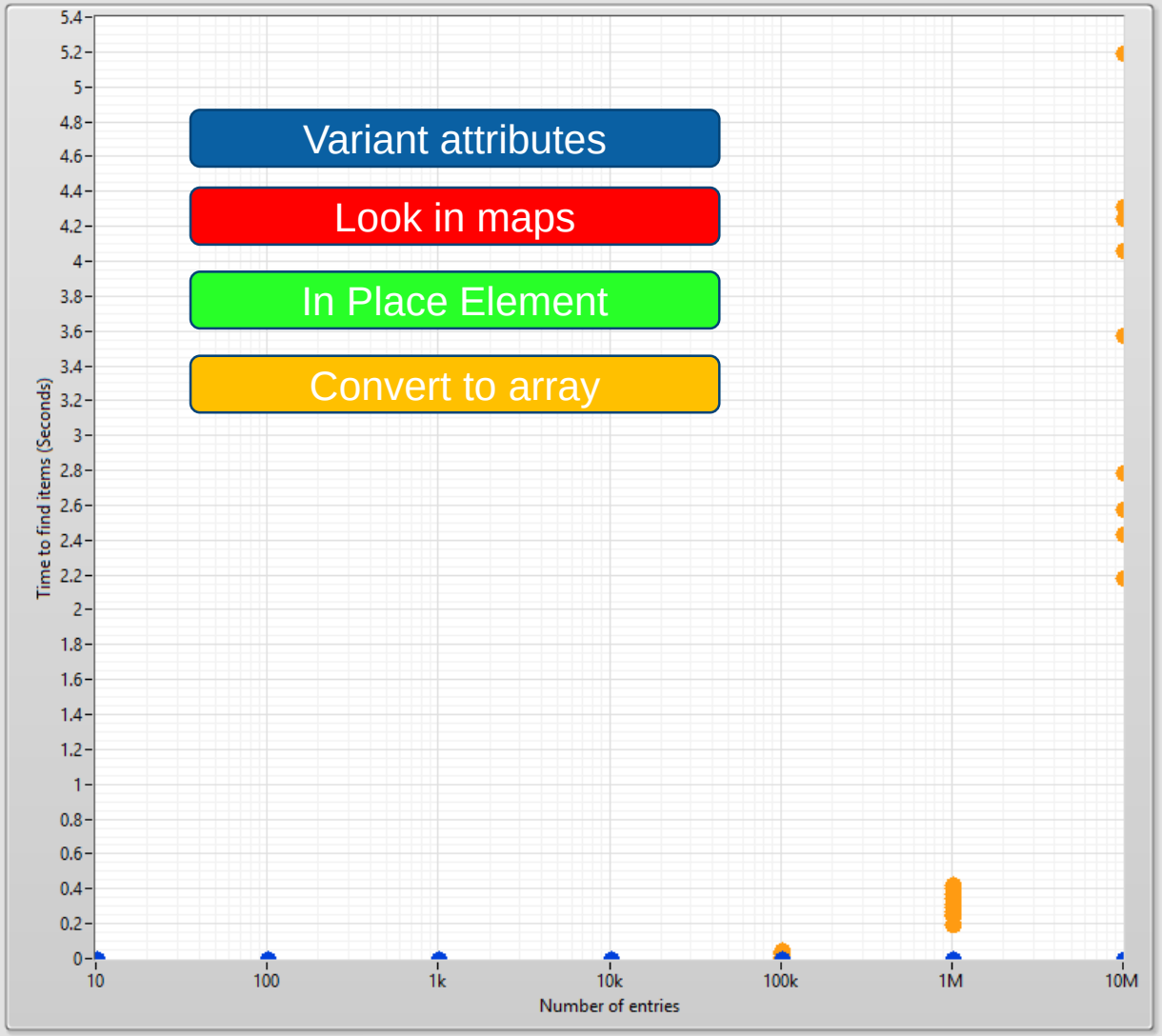
Benchmarking Maps.vi

File Edit View Project Operate Tools Window Help

Build table data



Find table items

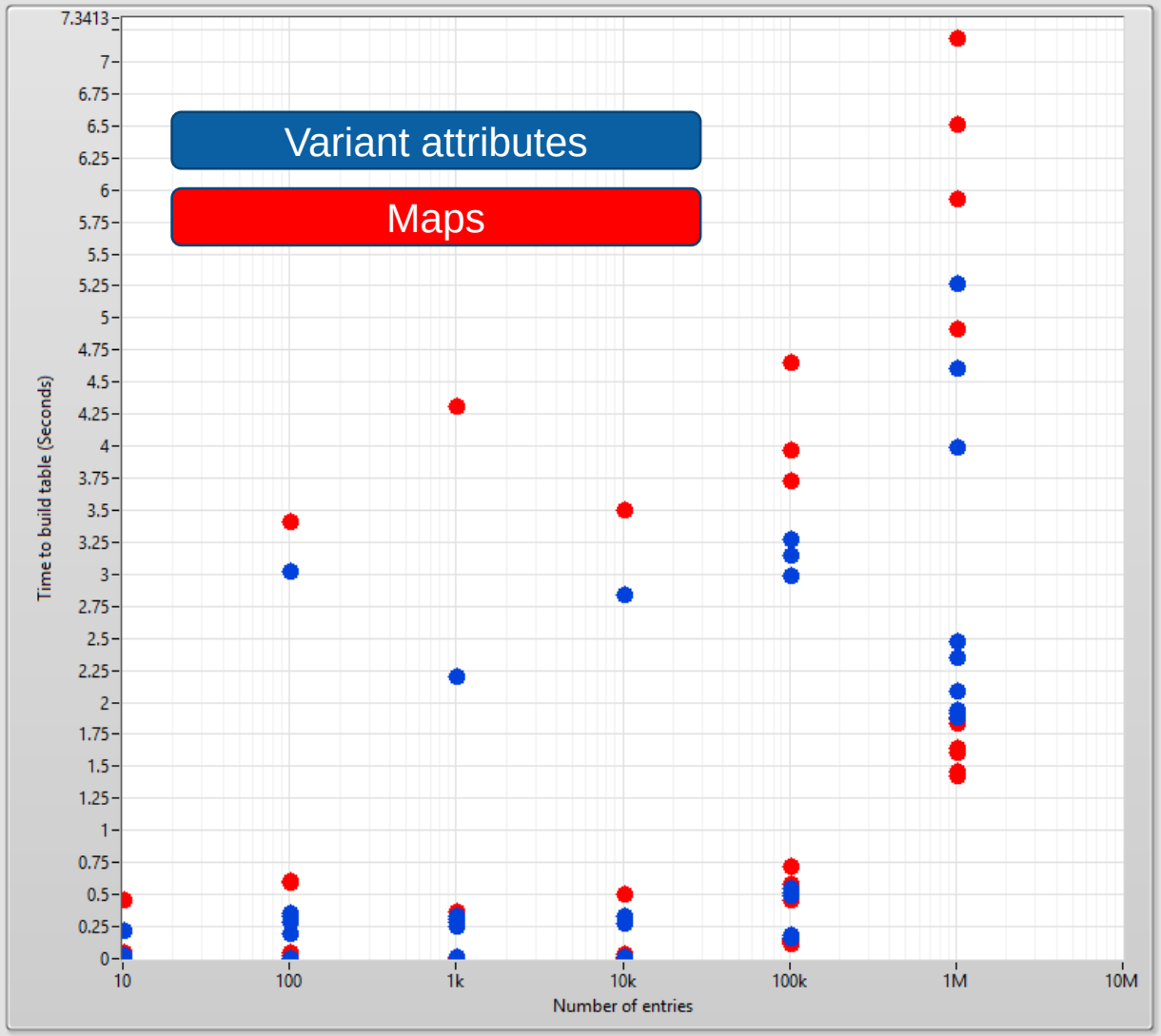


Examples.lvproj/My Computer

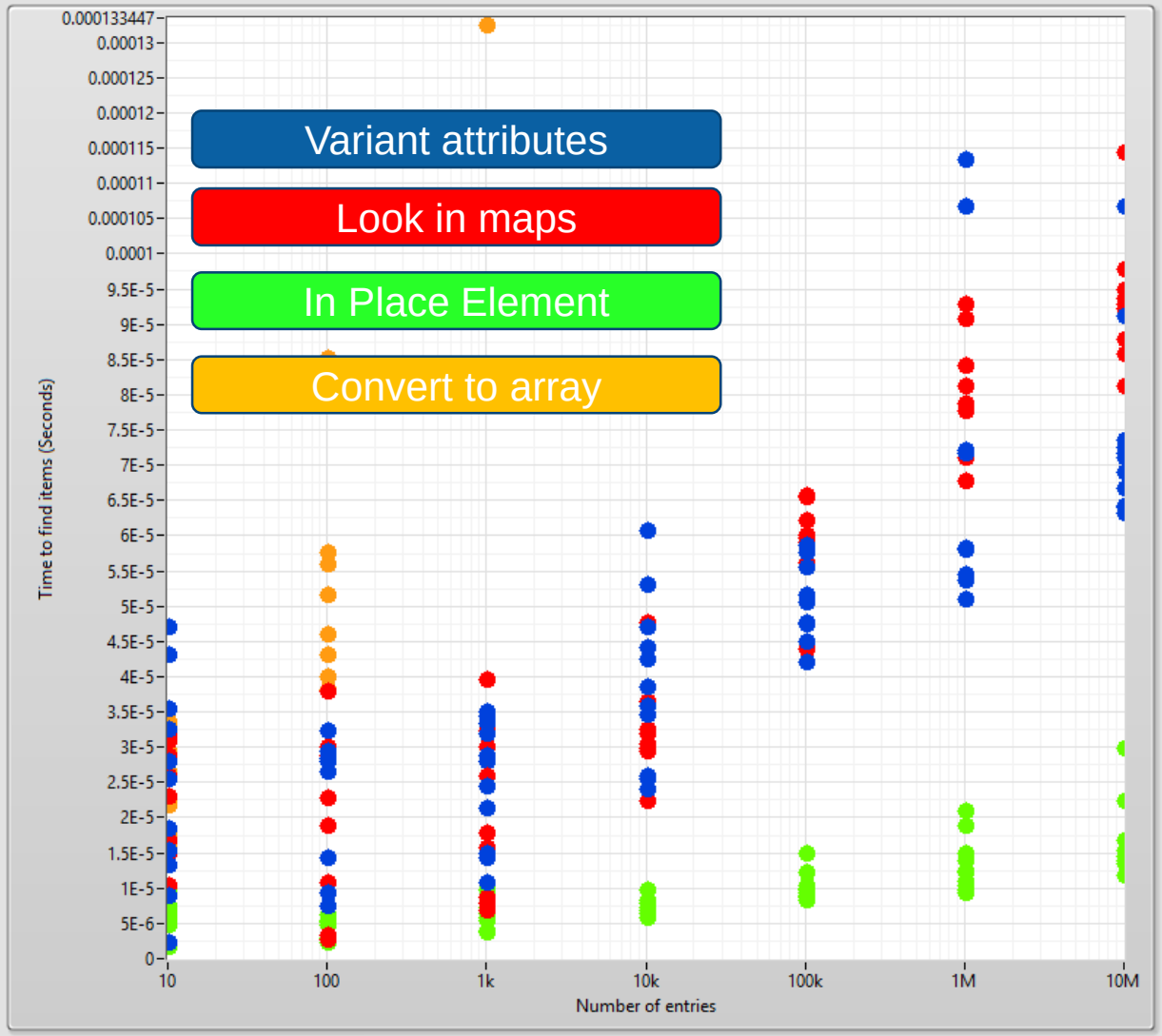
Benchmarking Maps.vi

File Edit View Project Operate Tools Window Help

Build table data



Find table items



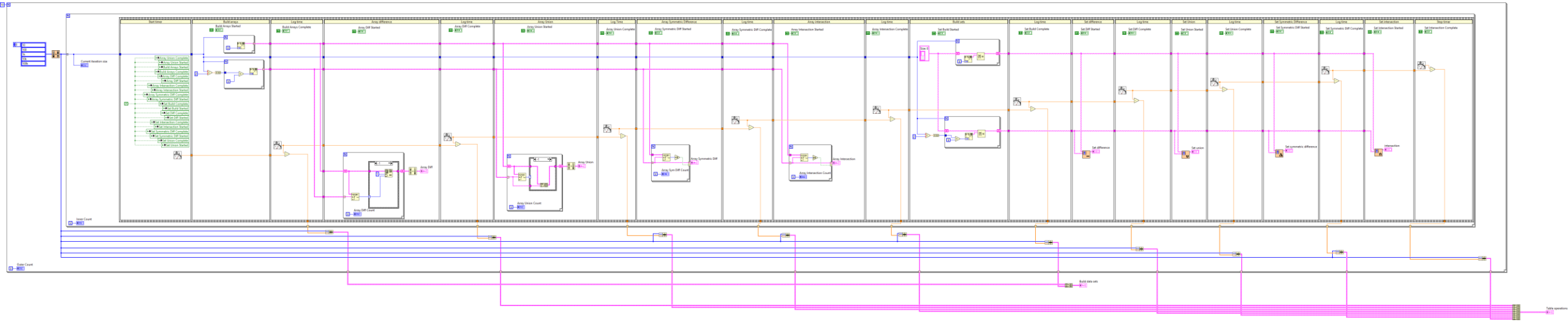
Examples.lvproj/My Computer

Performance – Sets vs arrays

Performing the same 5 operations on an array and then set

- Build data set
- Difference (elements in first set but not in second)
- Union (all elements that belong to either set)
- Symmetric Difference (elements in only one set)
- Intersection (elements in both sets)

Which is faster? And by how much???



NATIONAL INSTRUMENTS

Benchmarking Sets.vi

File Edit View Project Operate Tools Window Help

Build data sets

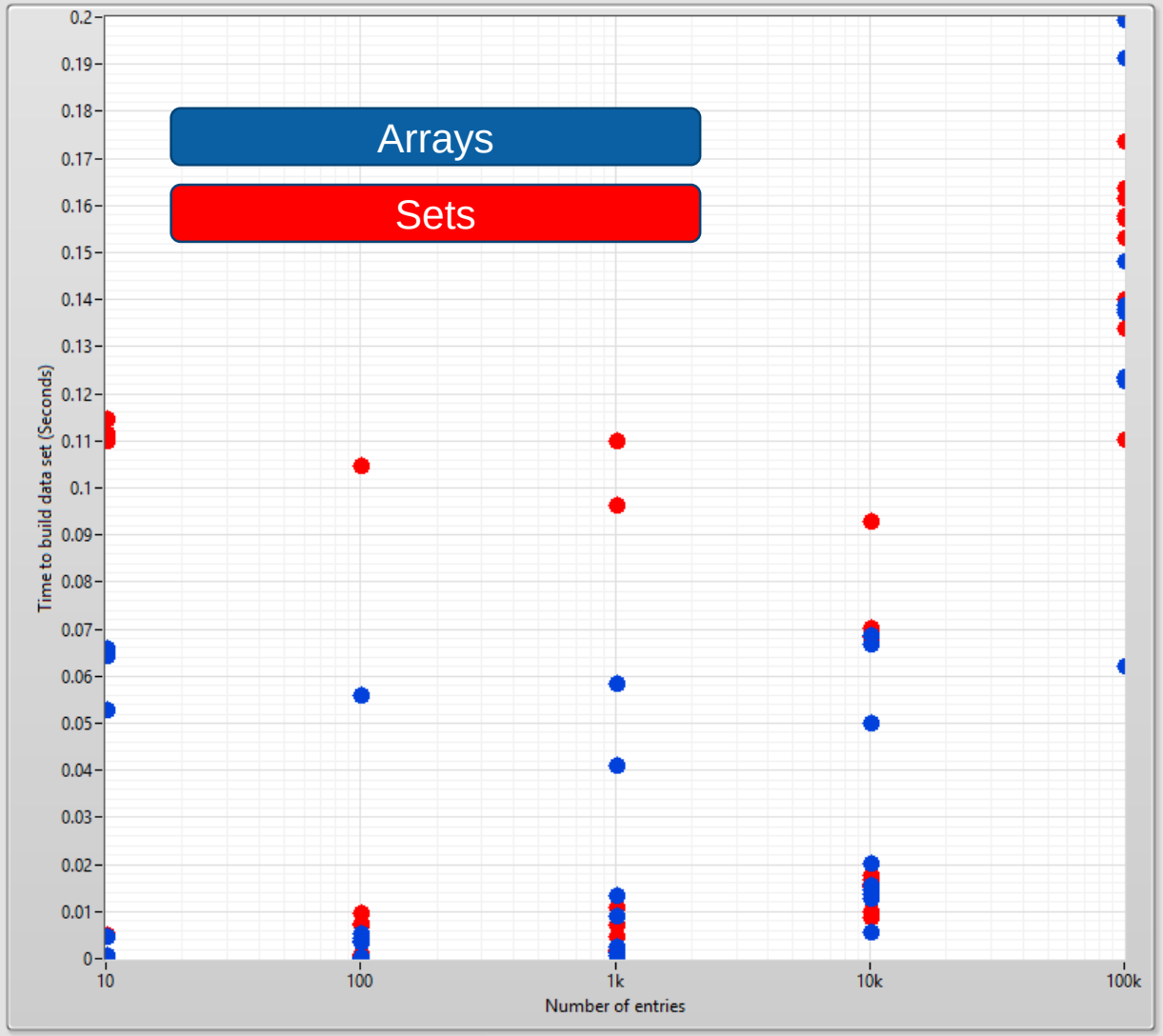
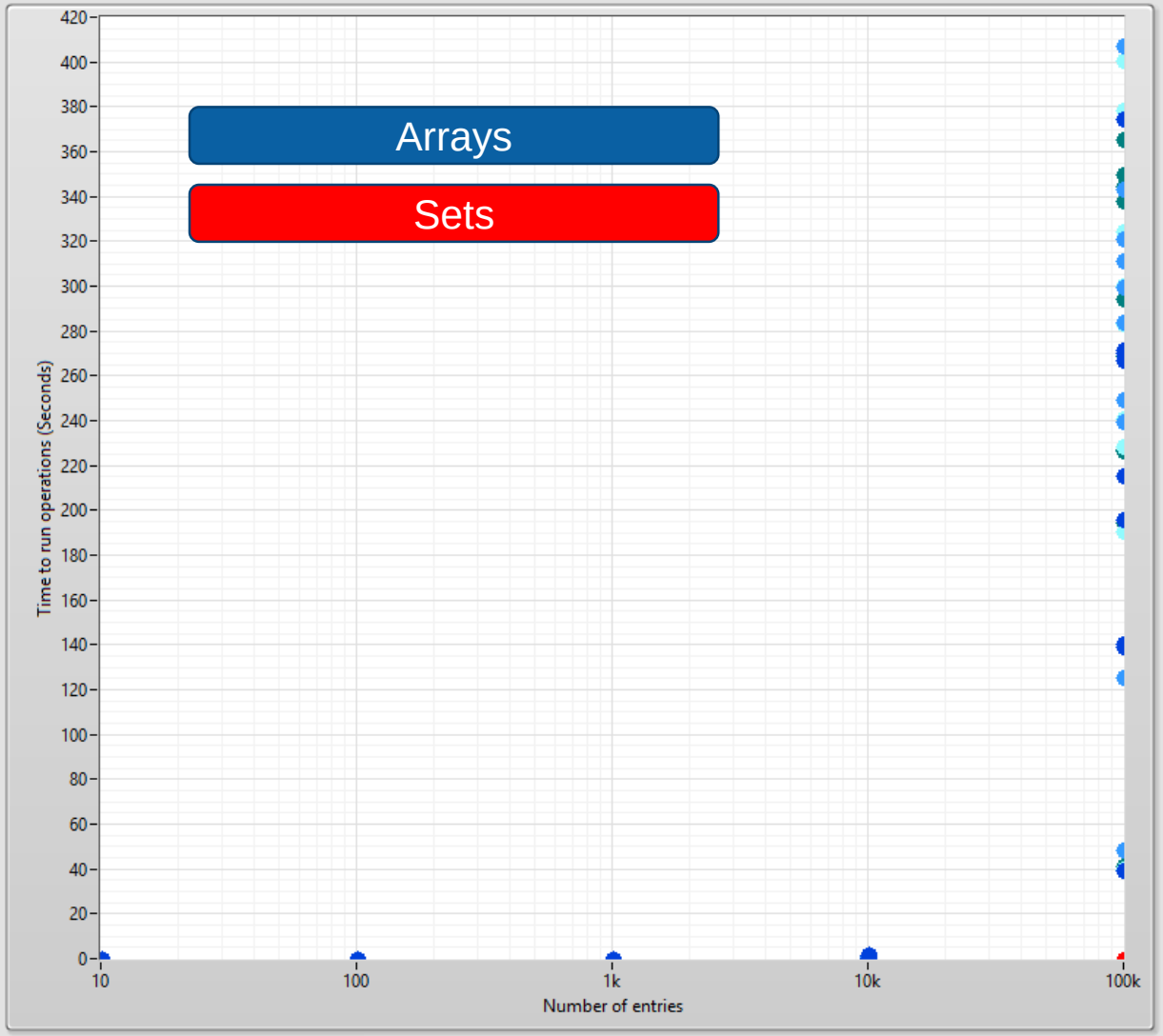
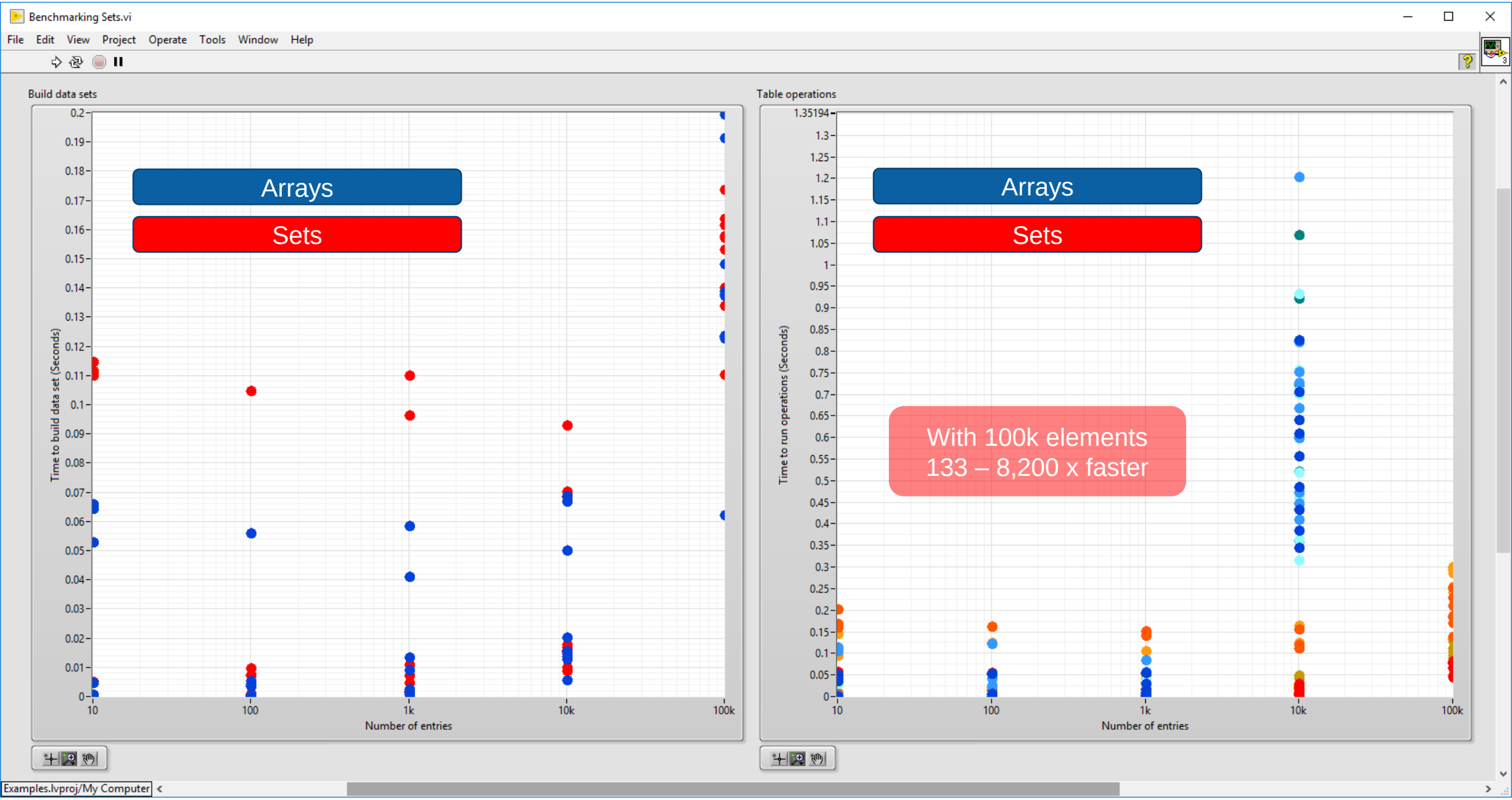


Table operations



Examples.lvproj/My Computer

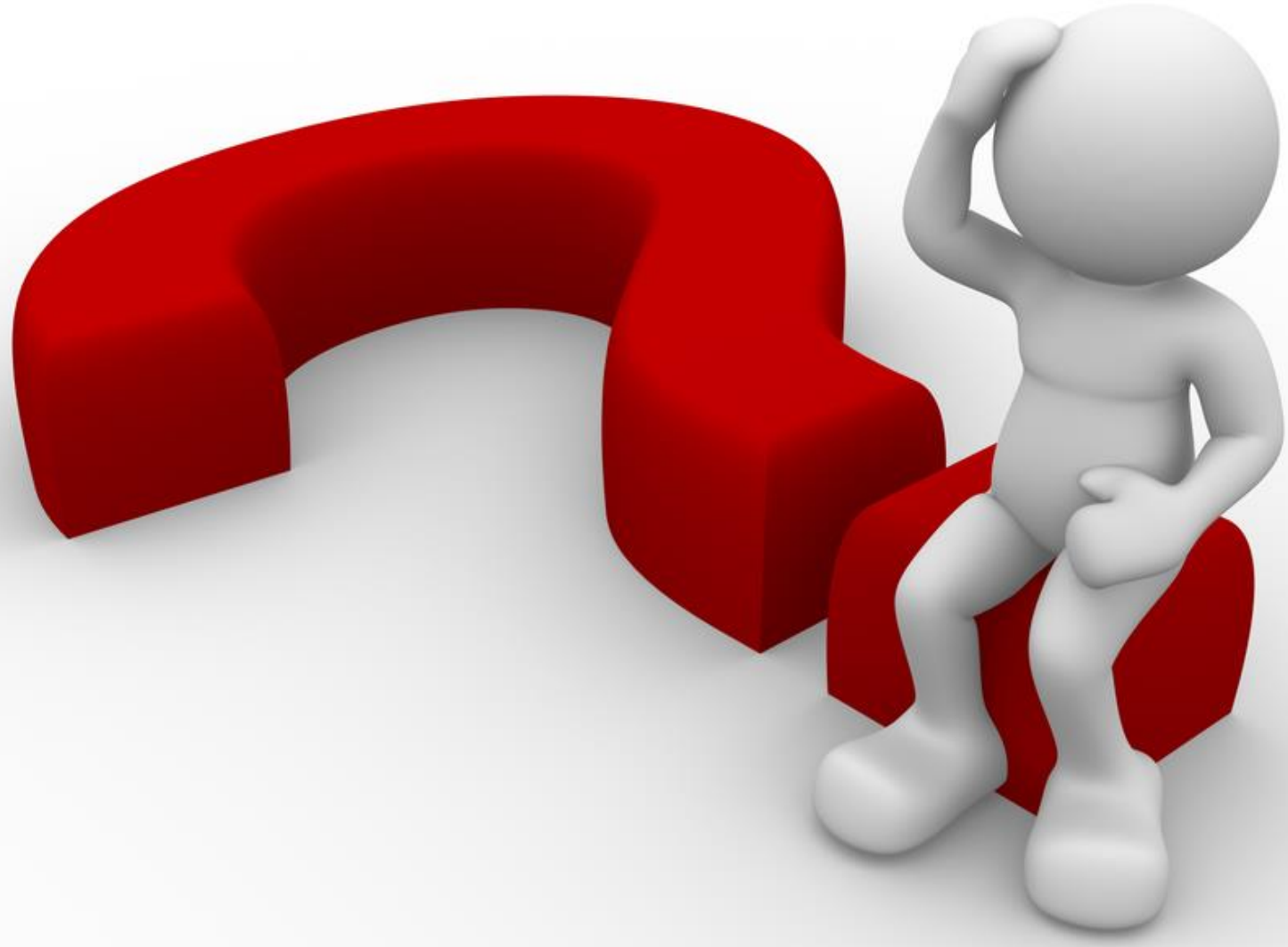

Benchmarking Sets.vi
File Edit View Project Operate Tools Window Help
Build data sets
Time to build data set (Seconds)
Number of entries
Arrays
Sets
Table operations
Time to run operations (Seconds)
Number of entries
Arrays
Sets
With 100k elements
133 – 8,200 x faster
Examples.lvproj/My Computer

Summary

- Sets and Maps are another tool in LabVIEW
  - Great when you need a flexible data table which enables quick/efficient data access
- Enables new ways of building algorithms
  - Should aid reducing complexity in some cases
  - Should be higher performance in some cases, especially very large data sets
- Arrays and variant attributes still remain as good solutions to some use cases

- Bottom line – Sets and maps are potentially a very powerful tool to understand and use in LabVIEW
  - Enables some very abstract but powerful programming methods – up to you if that is beneficial or not

https://github.com/PeteHorn/LV-Sets-Maps-UKTAG



Scan me