Automation and Make

Dr. Fang (Cherry) Liu
Scientific Computing Consultant
PACE, Georgia Tech

What is Make?

- **GNU Make** is a tool which controls the generation of executables and other non-source files of a program from the program's source files. **Make** gets its knowledge of how to build your program from a file called the makefile, which lists each of the non-source files and how to compute it from other (source) files.
- Make can be used to
  - Compile source code into executable programs or libraries (software development)
  - Run analysis scripts on raw data files to get data files that summarize the raw data
  - Run visualization scripts on data files
  - Parse and combine text file and plots to create papers

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Benefit by Using Make

- Can help to automate repetitive commands
- Save time
- Reduce the risk of making errors
- Ensure automatically-generated artifacts are only recreated when the files were used to create these have changed

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Example Code and Data

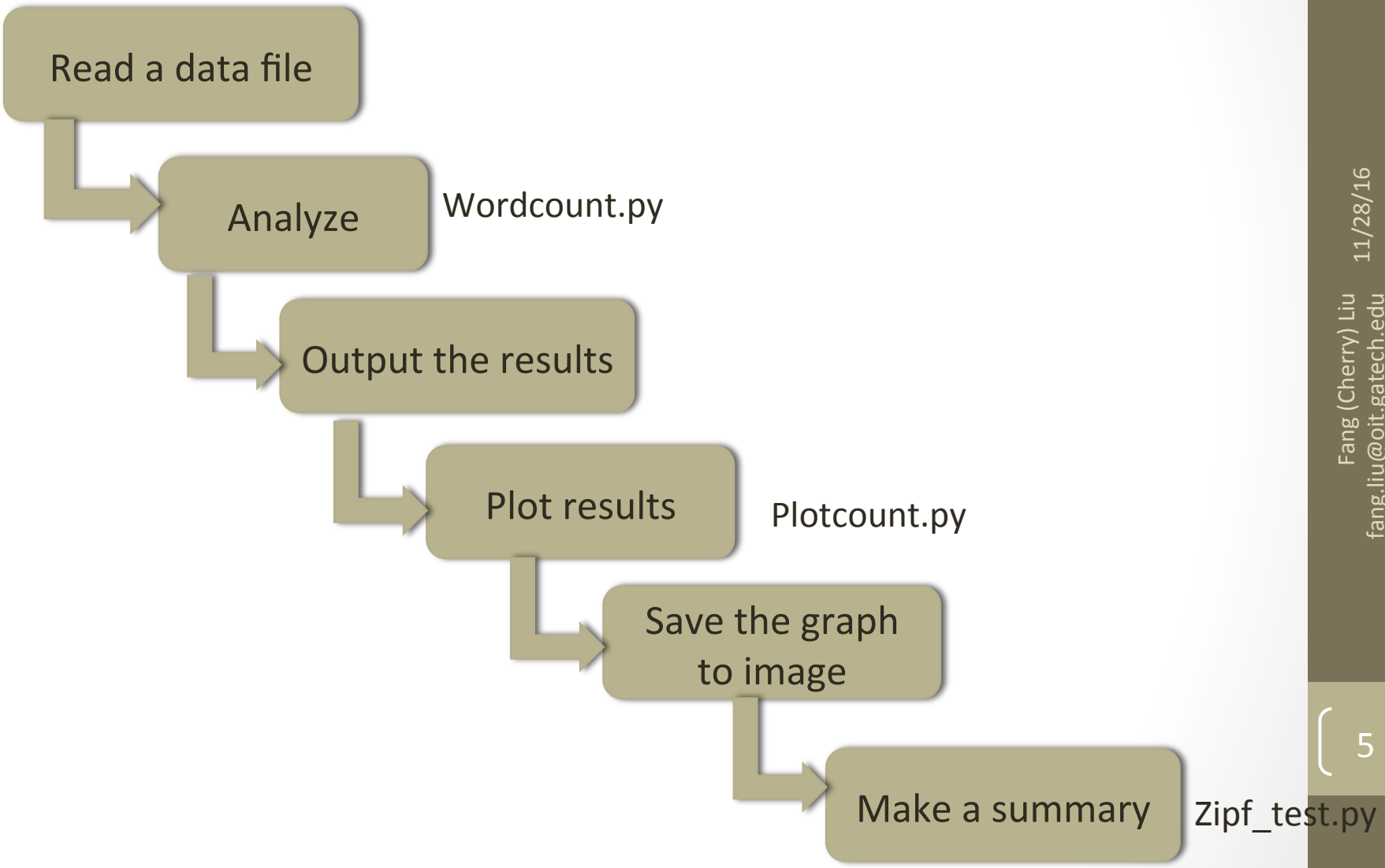
For the test example:
3 python files to processing data in ***books*** directory, and to test Zipf's Law:

The most frequently-occurring word occurs approximately twice as often as the second most frequent word.

```
|   -- .:
|   --  books
|   --  plotcount.py
|   --  wordcount.py
|   --  zipf_test.py
------------------------------
-----
| books:
|   --  abyss.txt
|   --  isles.txt
|   --  last.txt
|   --  sierra.txt
------------------------------
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Example Workflows



Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Approach Using Shell Script

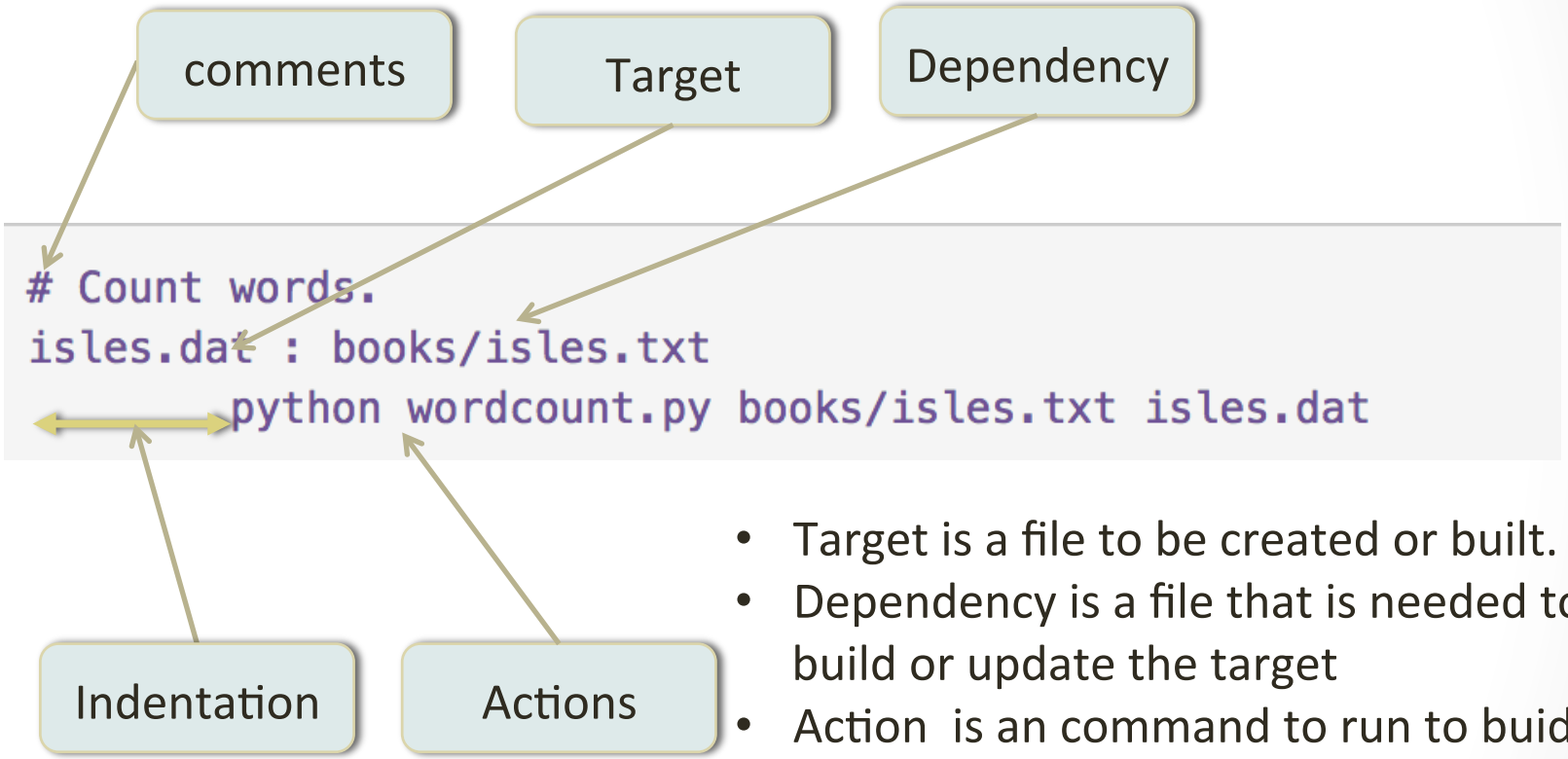
```
# USAGE: bash run_pipeline.sh
# to produce plots for isles and abyss
# and the summary table for the Zipf's law tests

python wordcount.py books/isles.txt isles.dat
python wordcount.py books/abyss.txt abyss.dat

python plotcount.py isles.dat isles.png
python plotcount.py abyss.dat abyss.png

# Generate summary table
python zipf_test.py abyss.dat isles.dat > results.txt
run_pipeline.sh (END)
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Approach using shell script (Cont.)

- Benefit:
  - Explicitly documents the pipeline
  - Allow reproducing the full analysis in one command
  - Prevent typos or mistakes
- Shortcomings:
  - Minor modifications on the code leads to rerun the full script

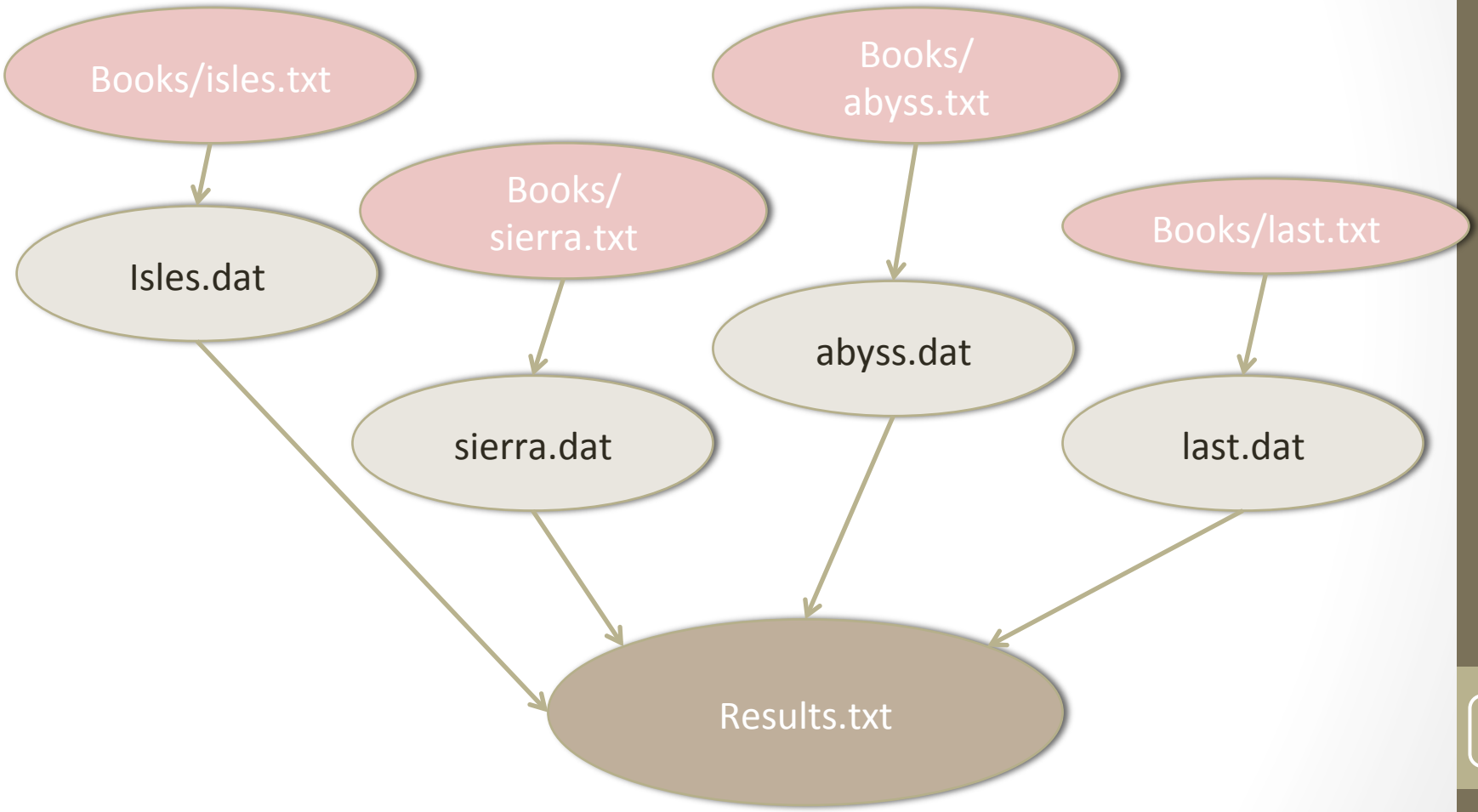
Simple Makefile



- Target is a file to be created or built.
- Dependency is a file that is needed to build or update the target
- Action is an command to run to buid or update target using dependencies.
- Run as: make <target>
- Indentation has to be TAB instead of space

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Data Dependency



Fang (Cherry) Liu
fang.liu@oit.gatech.edu

.PHONY target (Makefile)

- .PHONY target doesn't build anything,
- Use it to:
  - Avoid a conflict with a file of the same target name
  - Improve performance

```
.PHONY : clean
clean :
        rm -f *.dat
```

```
clean :
        rm -f *.dat
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Automatic Variables (Makefile. 1)

- To remove duplication in a Makefile
  - $@ : the target of the current rule
  - $^ : all the dependencies of the current rule
  - $< : the first dependency of the current rule

```
results.txt : isles.dat abyss.dat last.dat
	python zipf_test.py abyss.dat isles.dat last.dat > results.txt
```

Is the same as :

```
results.txt : isles.dat abyss.dat last.dat
	python zipf_test.py $^ > $@
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Dependencies on Data and Code

- updating a subset of the files in the pipeline triggers rerunning the appropriate downstream steps.
- Quiz:

```
$ touch books/last.txt
$ make results.txt
```

  - A. only last.dat is recreated
  - B. all .dat files are recreated
  - C. only last.dat and results.txt are recreated
  - D. all .dat and results.txt are recreated

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Pattern Rules (Makefile.2)

- Further reduce the repeated content by introducing a single pattern rule to build any .dat file from a .txt file in books/:

```
%.dat : books/%.txt wordcount.py
        python wordcount.py $< $*.dat
```

wildcard

Matches wildcard target

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Variables or Macros (Makefile. 3)

- Using variables to represent the scripts allows the easy switch in language or script name.

```
# Count words script.
COUNT_SRC=wordcount.py
COUNT_EXE=python $(COUNT_SRC)

# Test Zipf's rule
ZIPF_SRC=zipf_test.py
ZIPF_EXE=python $(ZIPF_SRC)
```

```
include config.mk

results.txt : *.dat $(ZIPF_SRC)
        $(ZIPF_EXE) *.dat > $@

.PHONY : dats
dats : isles.dat abyss.dat last.dat


%.dat : books/%.txt $(COUNT_SRC)
        $(COUNT_EXE) $< $*.dat
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Functions (Makefile.4)

- Use functions to write more complex rules
  - *wildcard* function : get lists of files matching a pattern

```
TXT_FILES=$(wildcard books/*.txt)
```

```
.PHONY : variables
variables:
	@echo TXT_FILES: $(TXT_FILES)
```

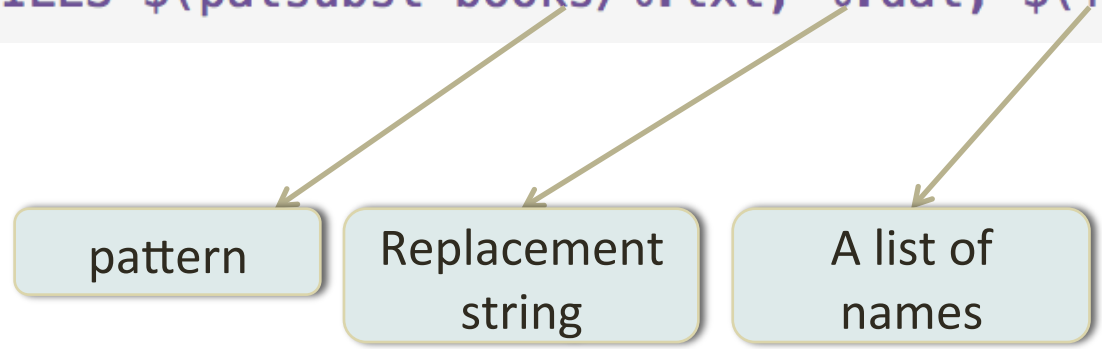
Only prints the result of echo

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Functions (Cont.)

- *patsubst* function : pattern substitution, takes a pattern, a replacement string and a list of names in order

```
DAT_FILES=$(patsubst books/%.txt, %.dat, $(TXT_FILES))
```



Self-Documenting Makefile

- by adding specially-formatted comments and a target to extract and format them.

```
## variables   : Print variables.
```

```
.PHONY : help
help : Makefile
	@sed -n 's/^##//p' $<
```

Fang (Cherry) Liu
fang.liu@oit.gatech.edu

Conclusion

- Automated build tools such as Make can help us:
  - Automate repetitive commands
  - Save us time and reduce the risk of making errors with manually ran commands
  - Save time by ensuring that automatically-generated artifacts are only recreated when the dependent files have changed
  - The notion of targets, dependencies and actions serve as a form of documentation, recording dependencies between code, scripts, tools, configurations, raw data, derived data, etc.

Acknowledgement

- This course is tailored based on software carpentry (http://software-carpentry.org/) material