The Robot Army

Distributed Processing for the Evil Genius

It is a BRUTE FORCE, DISTRIBUTED PROCESSING system that is uniquely EASY TO USE.

I’ll describe the underlying mechanism, some similar systems, and what makes ROBOT ARMY different.
The Robot Army

What is it?

- BRUTE FORCE
- DISTRIBUTED
- EASY TO USE
- COMMAND ORIENTED
The Robot Army

- What is the underlying idea?
- What is it similar to?
- How is it different?
The Mechanism

- transform
- partition
- aggregate

Our story begins a few years ago...
The Word Count Task

```
cat corp | tr -Cs [:alnum:] "\n" | sort | uniq -c > out
```
Implementations

- MapReduce
- Hadoop
- Disco
- CouchDB
all “code-oriented”

- Use the supported language
- Learn the API for that language
- Reuse existing serial programs?
  - Retrofit them – IF they are already in the supported language

These systems are all oriented to the level of code, not commands.
Advantages of having the library API are minimal

The system knows too much

all “code-oriented”
The Robot Army is different

- Willfully ignorant
- Lazy
- Apathetic
The Robot Army is different

- Command Oriented, not Code Oriented
- API: STDIN STDOUT STDERR
- Language Agnostic
Advantages

- Use your favorite language
- Reuse existing programs
  - Unix utilities
  - one-liners: perl, awk, sed, tr
  - existing code base investment
Advantages

- Develop programs locally
- Test programs locally
- Run programs distributed
Local vs Distributed

Word Count Task comparison:

cat corp | tr -Cs [:alnum:] "\n" | sort | uniq -c > out

tt mapreduce corp 'tr -Cs [:alnum:] "\n"' 'uniq -c' out
Example: casual use

Example: OCR measurement for TREC legal track, 61GB

Developed, ran, presented results during a meeting

Comments:

“Awesome!”

“Can we get one of those?”

“You’re scaring me”
Try Robot Army

Google Code project: ROBOTARMY

http://code.google.com/p/robotarmy
E Pluribus Machina