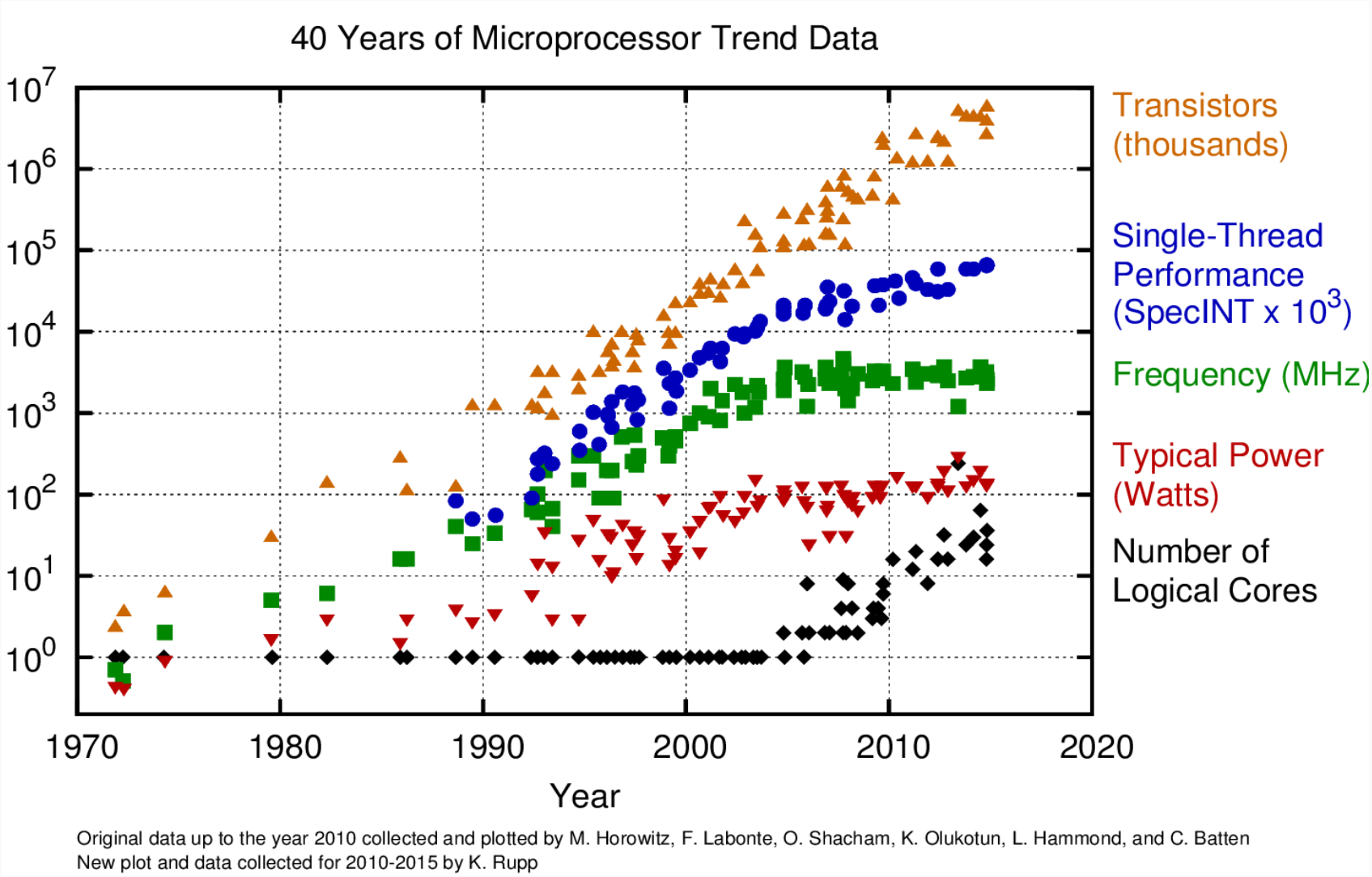
AJUSTE FINO AUTOMATIZADO UTILIZANDO COMPUTAÇÃO EM NUVEM

Pedro Bruel
phrb@ime.usp.br

29 de Setembro de 2015

Departamento de Ciência da Computação do IME, USP
MAC5910 - Programação para Redes de Computadores

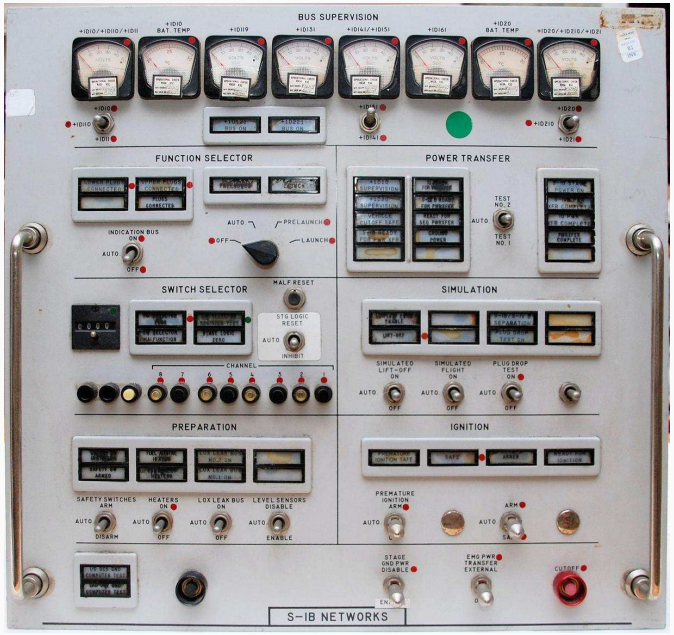
MOTIVAÇÕES



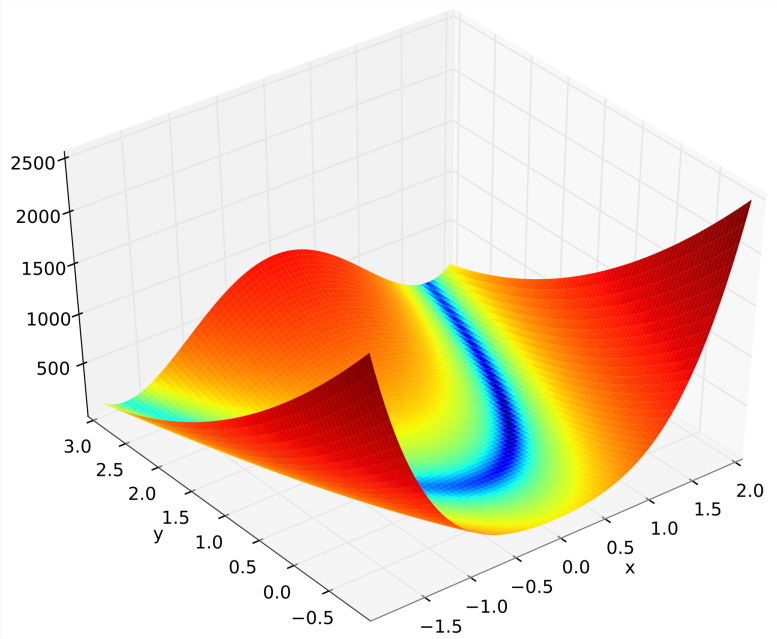
http://karlrupp.net/2015/06/40-years-of-microprocessor-trend-data

UMA SOLUÇÃO: *AUTOTUNING*

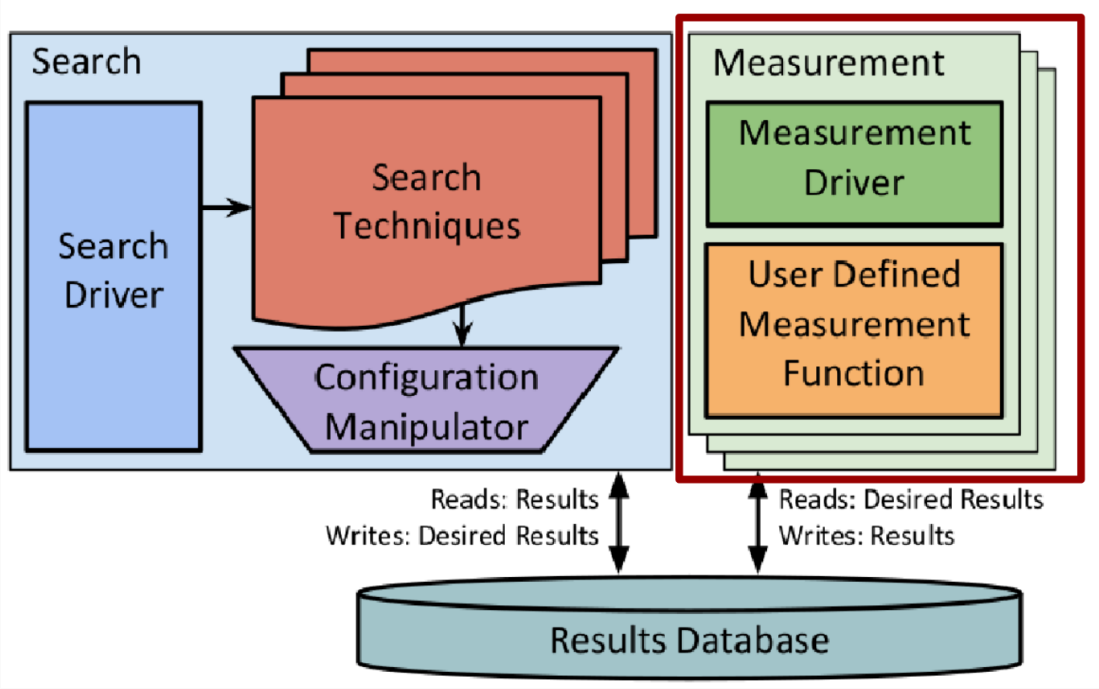
Configurações e Otimizações



Espaço de Busca



AUTOTUNING: OPENTUNER



Medições são sempre feitas localmente e sequencialmente

[2]Imagem: Ansel, Jason, et al. "Opentuner: An extensible framework for program autotuning."Proceedings of the 23rd ICPAC. ACM, 2014.

OBJETIVOS

Modificar o arcabouço OpenTuner para que seja possível realizar medições distribuídas na nuvem

RQ1: Como normalizar medições de desempenho feitas na nuvem?

RQ2: Para que tipo de problema é vantajoso utilizar os recursos da nuvem?

OBJETIVOS: LISTA

DONE:

- MeasurementClient
- MeasurementServer
- Interface com a Google Compute Engine
- Protocolo de Aplicação
- Avaliação de desempenho para o TSP, em alguns casos
- Correção de um bug no OpenTuner:
  https://github.com/jansel/opentuner/pull/73

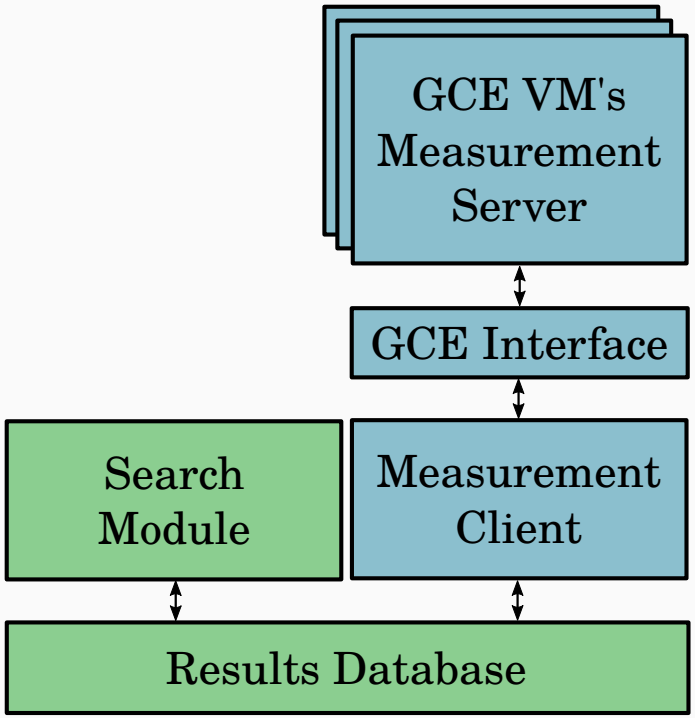
TODO:

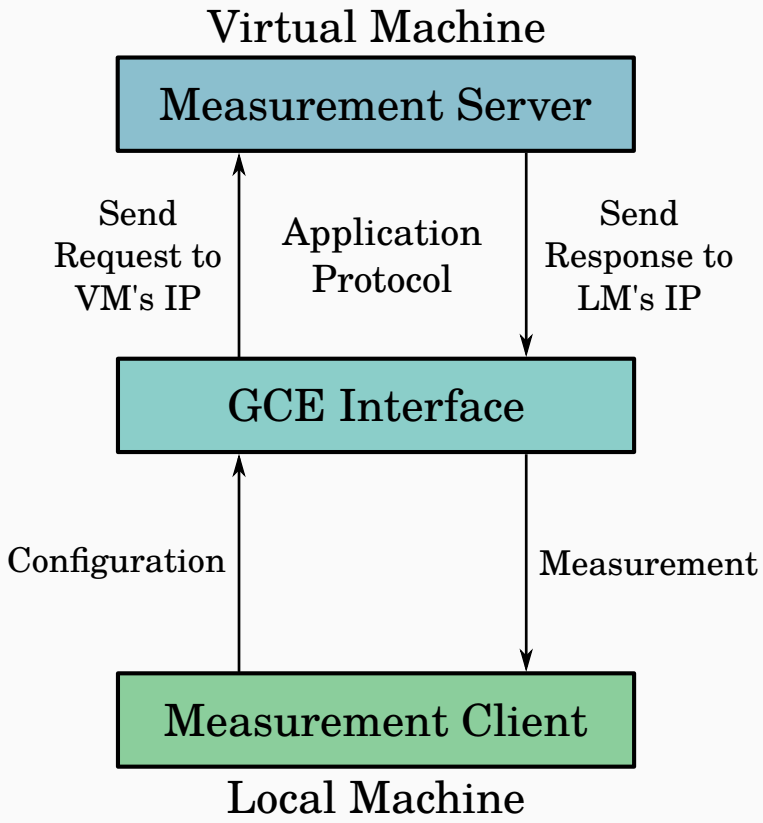
- Normalização de Resultados
- Composição do benchmark
- Avaliação de desempenho em mais problemas e casos

USANDO A GOOGLE COMPUTE ENGINE

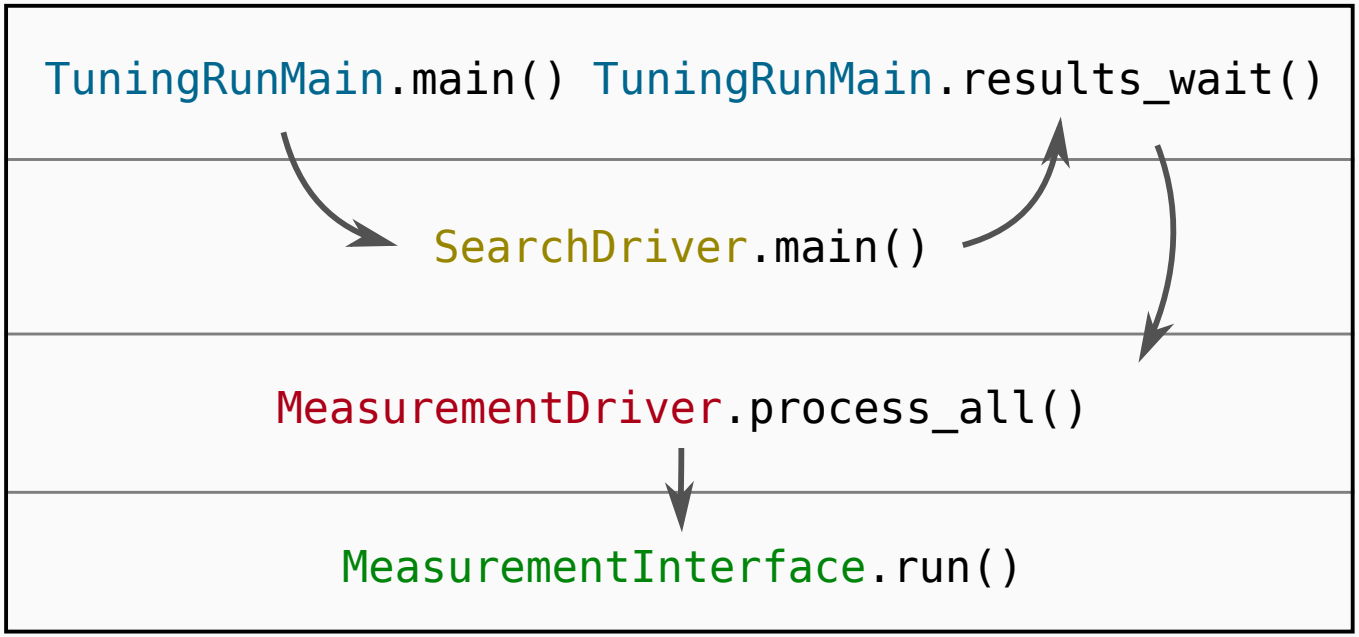
- Período de trial: US$300 por 60 dias
- Usei as máquinas de menor desempenho: apenas uma CPU
- Número limitados de CPUs e IPs por região: 23
- Construí uma imagem para acelerar a inicialização das VMs
- Dificuldades com os experimentos

CLIENTE E SERVIDOR

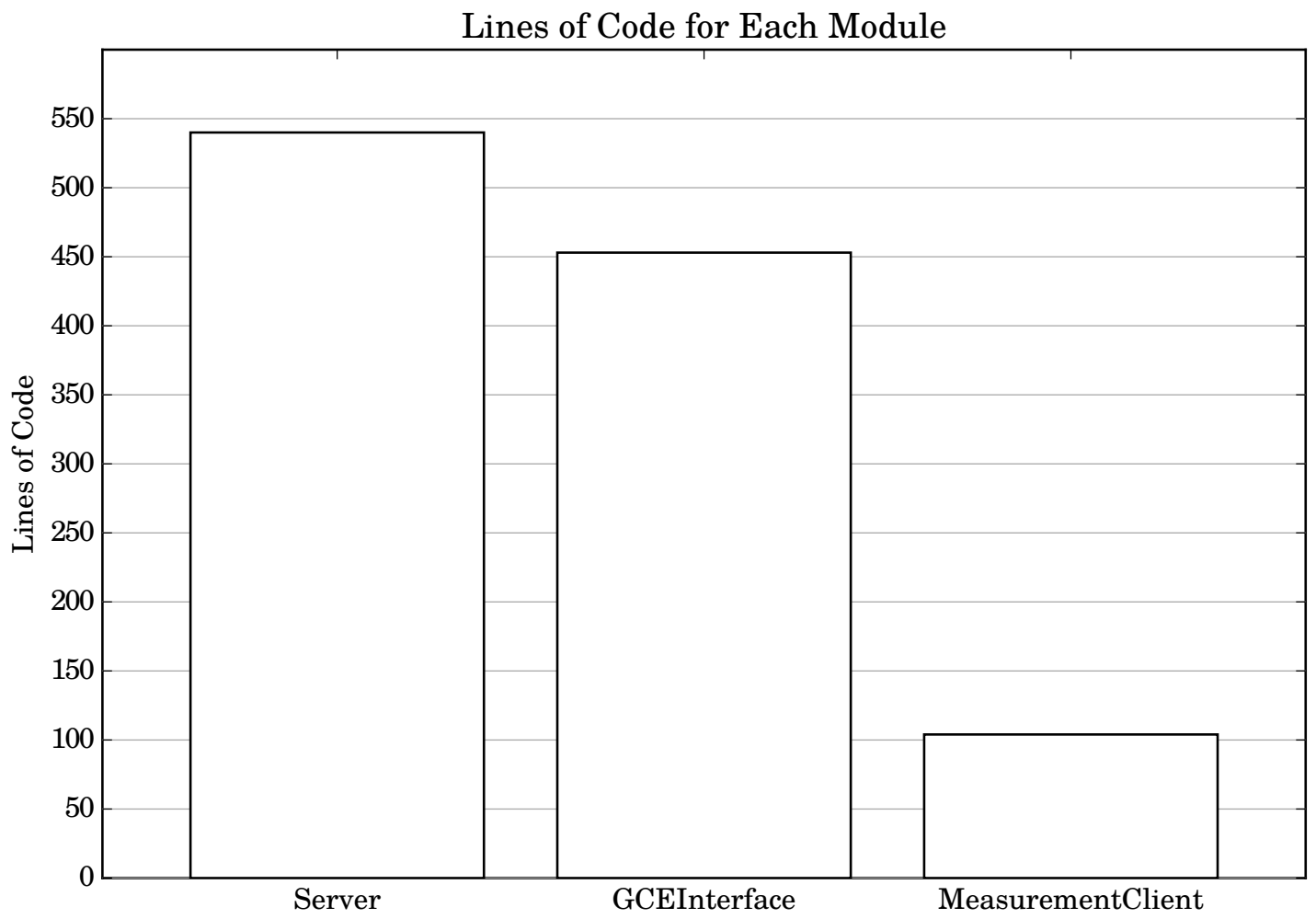




OpenTuner: fluxo de execução



ESFORÇO DE IMPLEMENTAÇÃO



SERVIDOR

Disponível sob GPL em: `https://github.com/phrb/measurement-server`

- Executado nas máquinas virtuais
- Espera por uma conexão TCP na porta 8080
- Espera por comandos
- Obtém o projeto do usuário à partir de um repositório git
- Importa a classe do usuário no servidor
- Executa medições utilizando o método `run` do usuário

PROTOCOLO: MENSAGENS

Command	Function	Message
START	Sets the server's status to AVAILABLE	"START"
STOP	Sets the server's status to STOPPED	"STOP"
STATUS	Requests the server current status	"STATUS"
DISCONNECT	Disconnects from the server	"DISCONNECT"
SHUTDOWN	Disconnects and shuts the server down	"SHUTDOWN"
CLONE	Clones a git repository to the virtual machine	"CLONE REPO_URL DIST_DIR"
LOAD	Imports the user's MeasurementInterface into the server	"LOAD TUNER_PATH INTERFACE_NAME"
MEASURE	Computes the measurement for a given configuration	"MEASURE CONFIG INPUT LIMIT"
GET	Requests a configuration's result	"GET RESULT_ID"

Tabela 1: Server messages.

PROTOCOLO: RESPOSTAS

Received Command	Response
START	“START ERROR_STATUS SERVER_STATUS”
STOP	“STOP ERROR_STATUS SERVER_STATUS”
STATUS	“STATUS ERROR_STATUS SERVER_STATUS”
DISCONNECT	“DISCONNECT ERROR_STATUS SERVER_STATUS”
SHUTDOWN	“SHUTDOWN ERROR_STATUS SERVER_STATUS”
CLONE	“CLONE ERROR_STATUS SERVER_STATUS [GIT_RET_CODE]”
LOAD	“LOAD ERROR_STATUS SERVER_STATUS”
MEASURE	“MEASURE ERROR_STATUS SERVER_STATUS [RESULT_ID]”
GET	“GET ERROR_STATUS SERVER_STATUS [RESULT_ID] [RESULT]”

Tabela 2: Server responses.

PROTOCOLO: RESPOSTAS NUMÉRICAS

Response	Code	Meaning
NO_ERROR	0	"START"
AVAILABLE	1	"STOP"
STOPPED	2	"STATUS"
UNAVAILABLE_ERROR	3	"STOP"
UNKNOWN_COMMAND_ERROR	4	"STATUS"
ARGUMENT_ERROR	5	"DISCONNECT"
GITCLONE_ERROR	6	"SHUTDOWN"
NO_FILE_ERROR	7	"CLONE REPO_URL DIST_DIR"
NO_RUN_METHOD_ERROR	8	"LOAD TUNER_PATH INTERFACE_NAME"
STARTED_ERROR	9	"MEASURE CONFIG INPUT LIMIT"
NOT_READY_ERROR	10	"GET RESULT_ID"
NO_SUCH_RESULT_ERROR	11	"GET RESULT_ID"

Tabela 3: Numeric responses.

INTERFACE

Disponível sob GPL em: https://github.com/phrb/gce_interface

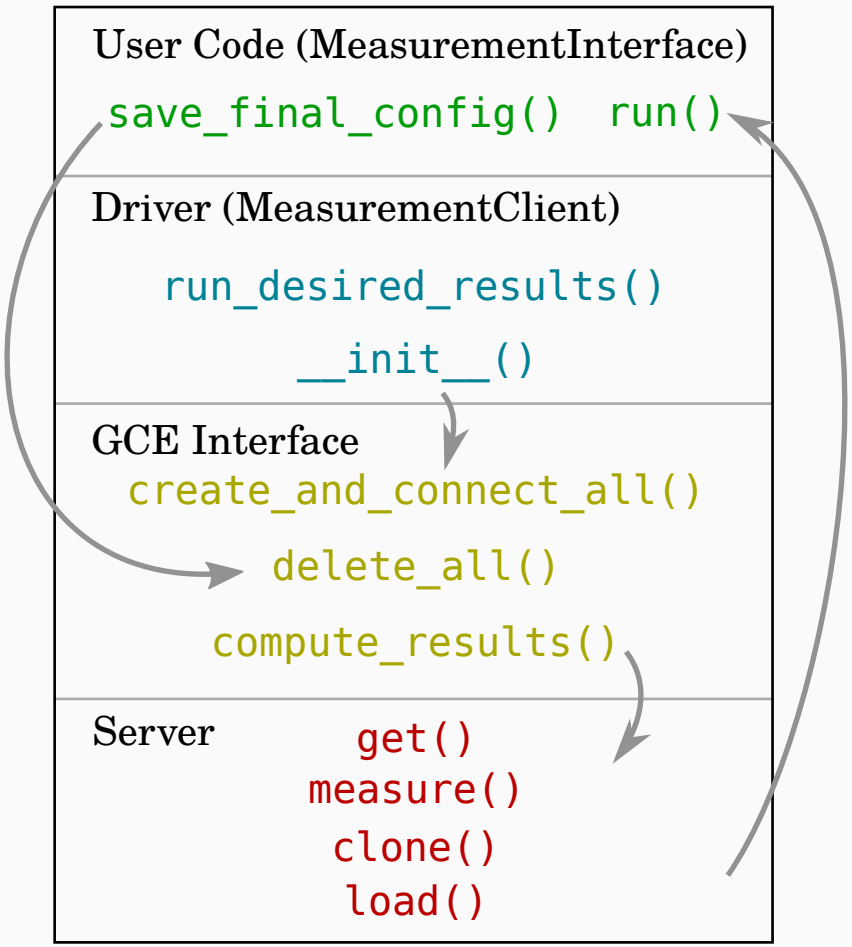
- Encapsula a comunicação com o servidor
- Serializa e deserializa os resultados e requisições usando o módulo pickle do Python
- Coordena a inicialização e término das máquinas virtuais

CLIENTE

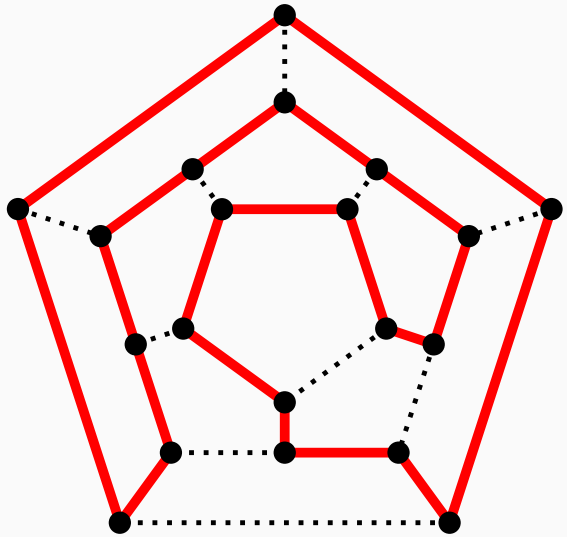
Disponível sob GPL em: https://github.com/phrb/measurement_client

- Processa requisições de resultados do OpenTuner
- Executa código do Usuário
- Faz requisições de resultados à interface

FLUXO DE EXECUÇÃO



EXPERIMENTOS: TRAVELLING SALESPERSON PROBLEM



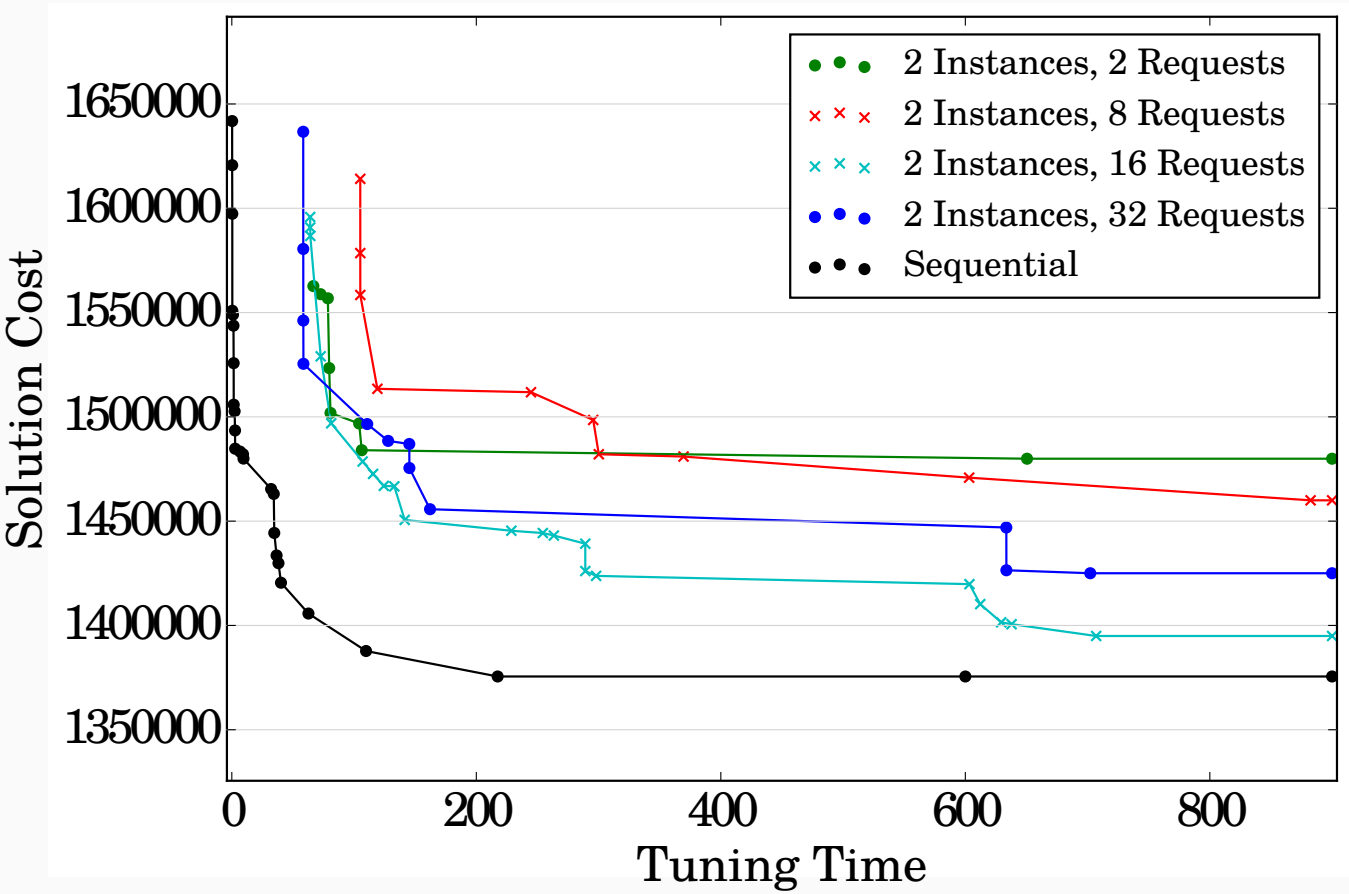
Encontrar o menor caminho que passa por todos os vértices de um grafo, voltando ao vértice de origem.

As soluções são representadas como permutações de cidades.

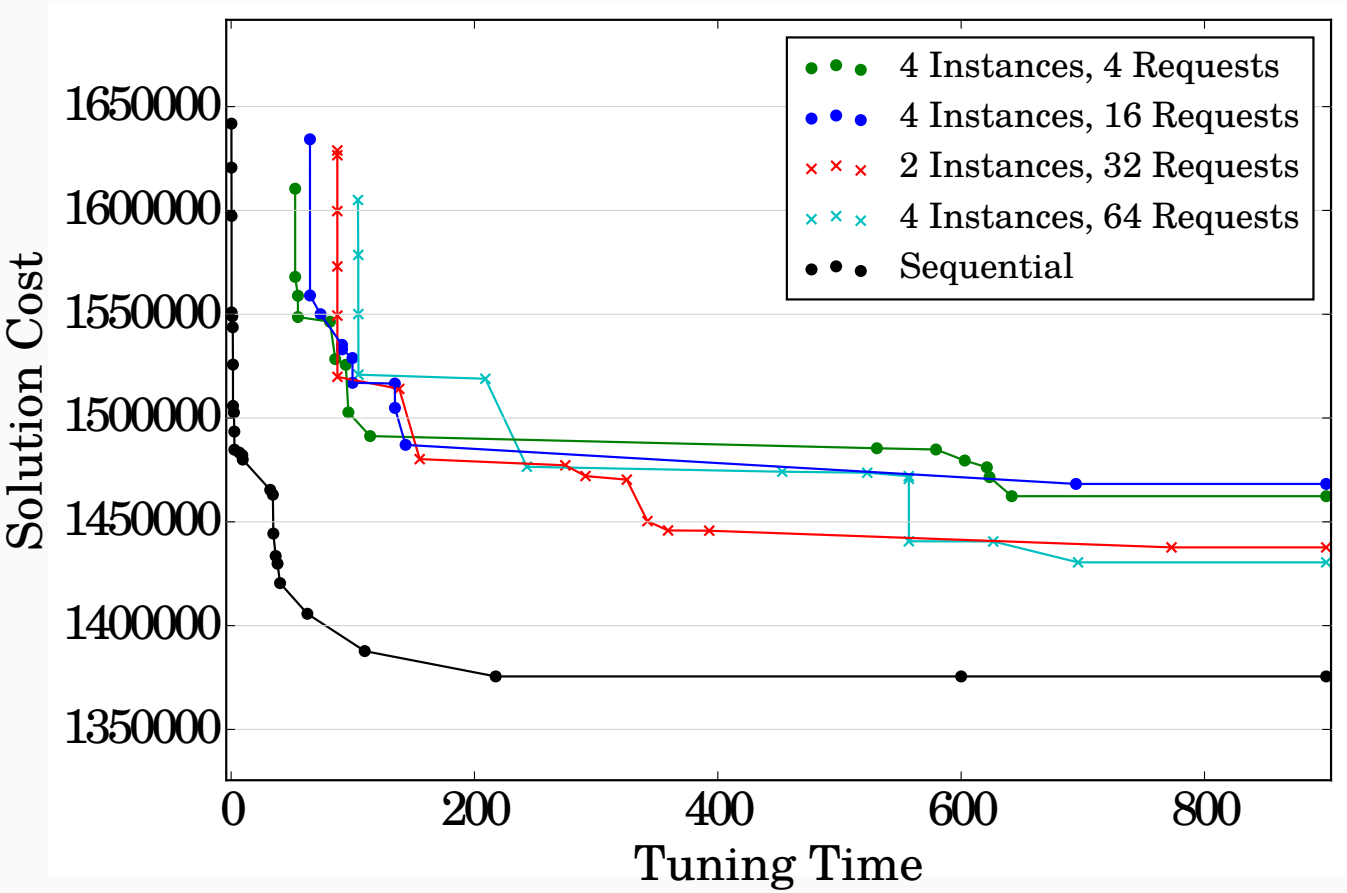
Avaliação do desempenho de um resolvedor de instâncias do TSP implementados com o OpenTuner.

Medições com números diferentes de máquinas virtuais e de requisições simultâneas.

TSP: RESULTADOS PARCIAIS



TSP: RESULTADOS PARCIAIS



NORMALIZAÇÃO DE RESULTADOS

RQ1: Como aplicar os resultados obtidos em máquinas virtuais diferentes da máquina local?

- Autotuning do modelo de desempenho
- Combinar resultados diferentes
- Simular a máquina local
- Executar o autotuner na nuvem

Obrigado!

AJUSTE FINO AUTOMATIZADO UTILIZANDO COMPUTAÇÃO EM NUVEM

Pedro Bruel
phrb@ime.usp.br

29 de Setembro de 2015

Departamento de Ciência da Computação do IME, USP
MAC5910 - Programação para Redes de Computadores