Sistemas Distribuídos

Análise de Desempenho

junho de 2015

A arte de avaliação de desempenho...

- Raj Jain. The Art of Performance Analysis. Wiley. 1991.

- metodologia
- cargas
- ferramentas

Comparando resultados

system	carga 1	carga 2
A	20	10
B	10	20

system	carga 1	carga 2	média
A	20	10	15
B	10	20	15

system	carga 1	carga 2	média
A	2	0.5	1.25
B	1	1	1

system	carga 1	carga 2	média
A	1	1	1
B	0.5	2	1.25

Avaliação de desempenho: abordagem sistemática (1)

1. definir qual o sistema estudado e os objetivos do estudo
   - fronteiras do estudo
   - o que se deseja estudar? o que se controla? etc
2. enumerar serviços e saídas
3. selecionar métricas
   - velocidade (throughput? latência? tempo de execução?), disponibilidade, acurácia, ...
4. enumerar parâmetros
   - características que afetam o desempenho
   - os que não iremos estudar devem permanecer estáveis
5. selecionar fatores para estudo
   - parâmetros a serem estudados
   - dificuldade com estudo simultâneo de muitos fatores e níveis

Avaliação de desempenho: abordagem sistemática (2)

6. selecionar técnica de avaliação
   - modelagem analítica
   - medidas experimentais
   - simulação
7. selecionar carga de trabalho
   - lista de requisições de serviço
8. projetar experimentos
9. analisar e interpretar resultados

Exemplo: comparação de RPC e pipes remotos (1)

1. sistema estudado: canal de comunicação
2. enumerar serviços: transferência de dados: "muitos" e "poucos"
3. selecionar métricas:
   - só operação correta
   - recursos envolvidos: máquina local (cliente), máquina remota (servidor), rede
   - tempo decorrido por transferência
   - taxa de transferência
   - tempo de CPU local por transferência
   - tempo de CPU remota por transferência

Exemplo: comparação de RPC e pipes remotos (2)

4. enumerar parâmetros:
   - velocidade CPUs
   - velocidade rede
   - sistema operacional
   - confiabilidade da rede
   - tempo entre transferências
   - número e tipo de dados
   - cargas externas (CPU e rede)
   - os que não iremos estudar devem permanecer estáveis

Exemplo: comparação de RPC e pipes remotos (3)

5. selecionar fatores:
   - tipo do canal: RPC X pipes
   - velocidade da rede: local X remota
   - tamanho dos dados transmitidos: pequeno X grande
   - número de chamadas consecutivas

Exemplo: comparação de RPC e pipes remotos (4)

6. técnica de avaliação
7. carga de trabalho
   - programa sintetizado para gerar as requsições especificadas
8. desenho do experimento
9. análise de dados

Técnicas de Avaliação

- modelagem analítica
- medidas experimentais
- simulação

- uso complementar

Limitações frequentes

- análise de complexidade do algoritmo utilizado
  - e para números pequenos?
  - distância entre algoritmo e implementação
- extrapolações a partir de medidas
  - escalabilidade

Uso complementar de medidas e análise

- análise inicial pode fornecer valores básicos para serem usados em modelo
- depois da construção do protótipo, o desenvolvedor pode confrontar o modelo com medidas
- ... restorna a medidas, à análise da implementação, ao modelo, ...

- até uma análise "no guardanapo" pode ajudar muito!

Desenho do Experimento

- projeto de casos a serem executados
- também pode se beneficiar de processo iterativo
- cuidados com forma de disparo e ordem de execução
- uso de ferramental estatístico

Motivos para surpresas

- programas com comportamento variável
- timer com acurácia insuficiente
- custos de inicialização e fechamento
- interferência de outros programas
- contenção

Técnicas de Avaliação

critério	mod. analítica	simulação	medidas
estágio	qq	qq	pós-protótipo
tempo	pequeno	médio	variável
ferramentas	analistas	LPs	intrumentação
acurácia*	baixa	moderada	variável
custo	baixo	médio	alto

Métricas

enorme variação...

- tempo total de execução
- taxa de atendimento
- latência
- tempo máximo de atendimento de uma requisição
- requisitos de memória
- custos de implementação
- custos de manutenção
- portabilidade
- escalabilidade

Cargas de trabalho

- cargas sintéticas e reais
- programas sintéticos e reais
- núcleos de processamento ou comunicação
  - criação de conteúdo dinâmico
  - mensagens em *ping-pong*
- aplicações exemplo
  - servidor http!
- programas geradores de carga
  - httperf
  - outros geradores: distribuições probabilísticas, etc
    - A. Beitch et alii. Rain: A Workload Generation Toolkit for Cloud Computing Applications - Technical Report UCB/EECS-2010-14.
  - geração de carga em outros programas interativos como jogo:
    - Halvorsen, Stig Magnus. Load Generation for Investigating Game System Scalability. Master thesis, University of Oslo, 2014

Ferramentas para medidas

- chamadas ao sistema ou a bibliotecas
- profilers
- ferramentas embutidas em geradores de carga

Ao medir tempo...

- acurácia de timers: uso de loops para alcançar patamares com alguma confiabilidade
- variação entre execuções: uso de várias repetições e registro de variância
  - ferramental estatístico para comparações: médias simples não bastam
- culturas diferentes em comunidades diferentes: exemplo livro Peter Pacheco

exemplo — medidas de tempo

Controller	Threads	Min.	Max.	Average	Standard Deviation
SRPT	8	1.73	224.90	113.08	64.66
SRPT	16	3.20	224.50	113.27	64.20
SEDA	16	1.53	224.32	113.44	64.69
Workstealing	16	1.69	259.22	117.07	66.78
Workstealing	8	1.47	276.69	124.55	71.46
SEDA_d	8/16	1.53	225.34	130.82	70.26
Leda	8	2.25	257.69	167.85	36.16
Leda	16	3.64	256.86	171.04	51.62
SEDA	8	1.25	393.79	199.12	113.95

Tabela: tempo de resposta no cliente (em segundos)

Medidas de tempo

- e em geral, o que querem dizer esses tempos...?

- diferenças da mesma ordem muitas vezes podem ser geradas por opções de compilador!
- alterações mínimas na arquitetura e instalação podem mudar resultados completamente
- ... então como usar esses resultados?

```
printf("The elapsed time = %e seconds\n", global_elapsed);
```

Here, we first execute a **barrier** function that approximately synchronizes all of the processes/threads. We would like for all the processes/threads to return from the call simultaneously, but such a function usually can only guarantee that all the processes/threads have started the call when the first process/thread returns. We then execute the code as before and each process/thread finds the time it took. Then all the processes/threads call a global maximum function, which returns the largest of the elapsed times, and process/thread 0 prints it out.

We also need to be aware of the *variability* in timings. When we run a program several times, it's extremely likely that the elapsed time will be different for each run. This will be true even if each time we run the program we use the same input and the same systems. It might seem that the best way to deal with this would be to report either a mean or a median run-time. However, it's unlikely that some outside event could actually make our program run faster than its best possible run-time. So instead of reporting the mean or median time, we usually report the *minimum* time.

Running more than one thread per core can cause dramatic increases in the variability of timings. More importantly, if we run more than one thread per core, the system will have to take extra time to schedule and deschedule cores, and this will add to the overall run-time. Therefore, we rarely run more than one thread per core.

Finally, as a practical matter, since our programs won't be designed for high-performance I/O, we'll usually not include I/O in our reported run-times.

2.7 PARALLEL PROGRAM DESIGN

So we've got a serial program. How do we parallelize it? We know that in general we need to divide the work among the processes/threads so that each process gets roughly the same amount of work and communication is minimized. In most cases, we also need to arrange for the processes/threads to synchronize and communicate

Relatos de resultados

- gráficos e tabelas
  - “ilusões”

Entendimento e repetibilidade

- "If you torture the data long enough, they will confess."

- movimento por repetibilidade
  - entendimento de resultados
- Open Science Foundation e outros completamente

Testbeds e portais como facilitadores

- manutenção de infraestruturas e dados

exemplos

- Planet Lab
- Forge, Geni, Fibre., ... (Internet do Futuro)
- portais científicos

e as métricas que não dizem respeito a tempos de computação/comunicação?

- curva de aprendizado
- tempo de desenvolvimento
- facilidade de escrever programas corretos
- tempo de manutenção

subjetivas?

- e quando somos totalmente objetivos?