

EFFICIENT EXTERNAL TABLE REORGANIZATION

Walt Burkhard

Computer Science and Engineering Department
University of California, San Diego



- Related work.
- External hashing.
- Performance: analysis & simulation.
- Conclusions and future work.



- Efficient Ordering of Hash Tables. 1977
- Uniform Hashing. 1983
- Hashing with Separators. 1987
- Double Hashing with Multiple Passbits. 2003
- Efficient Reordering of External Tables ...

Efficient Reordering of External Tables I



Example table comprised of buckets ... with $b = 3$ slots per bucket.

DOG
CAT

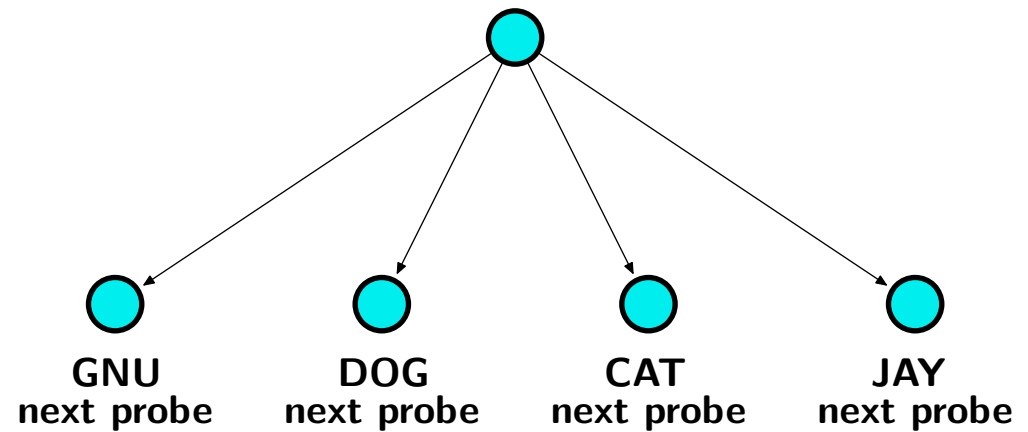
DOG
CAT
JAY

Double hashing scheme: h_1 & h_2



Insertion of GNU probes this bucket –

DOG
CAT
JAY



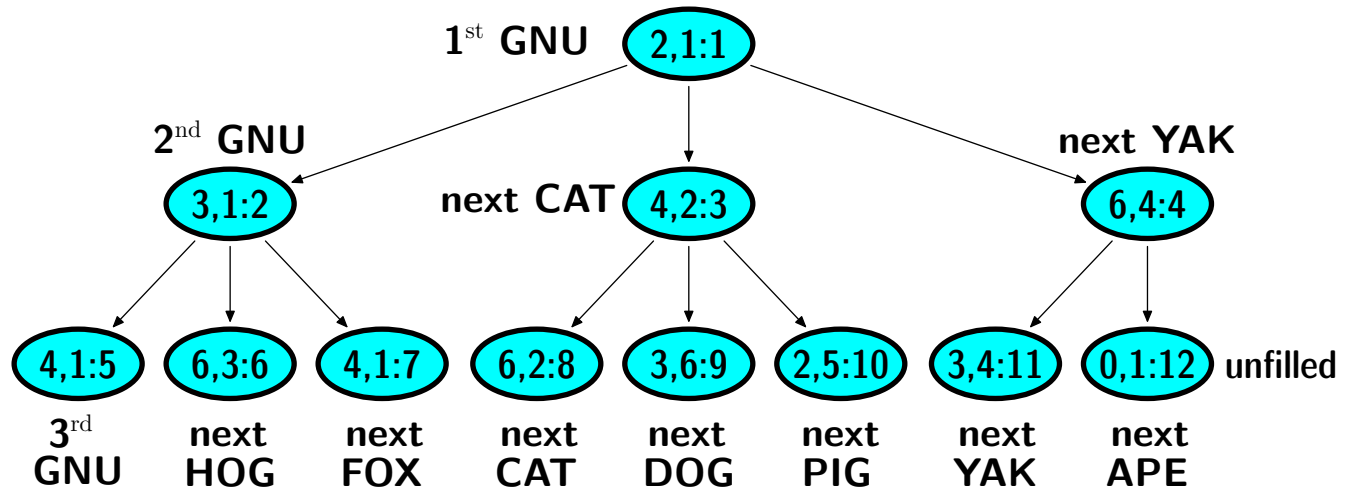
tree is explored but not explicitly built.

Efficient Reordering of External Tables III



0	1	2	3	4	5	6
		YAK	FOX	PIG		EEL
JAY		CAT	HOG	DOG		APE

	start	step
CAT	2	2
DOG	4	6
APE	6	1
PIG	4	5
HOG	3	3
YAK	2	4
JAY	0	4
EEL	6	2
FOX	3	1
GNU	2	1

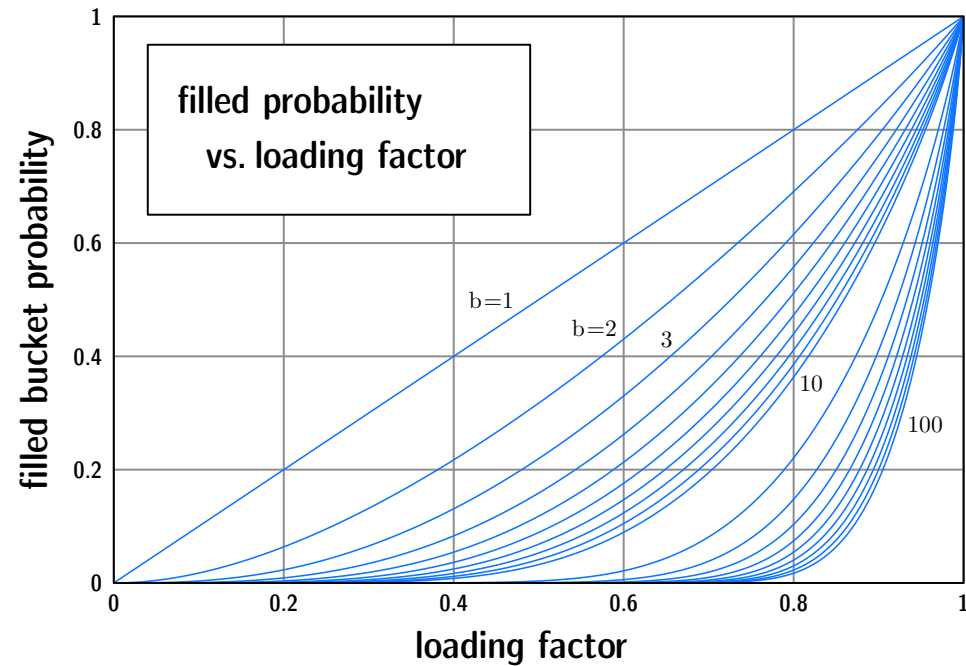




Filled bucket probability

$$\beta(\alpha) = q_b(\alpha).$$

$q_i(\alpha)$ is the probability a bucket contains i records for a table with loading factor $\alpha = m/(bn)$ for $0 \leq i \leq b$.



$$\frac{dq_0}{d\alpha} = -b \times q_0(\alpha) / (1 - q_b(\alpha))$$

$$\frac{dq_i}{d\alpha} = b \times (q_{i-1}(\alpha) - q_i(\alpha)) / (1 - q_b(\alpha)) \quad 0 < i < b$$

$$\frac{dq_b}{d\alpha} = b \times q_{b-1}(\alpha) / (1 - q_b(\alpha)).$$



- **Successful search length** L_S

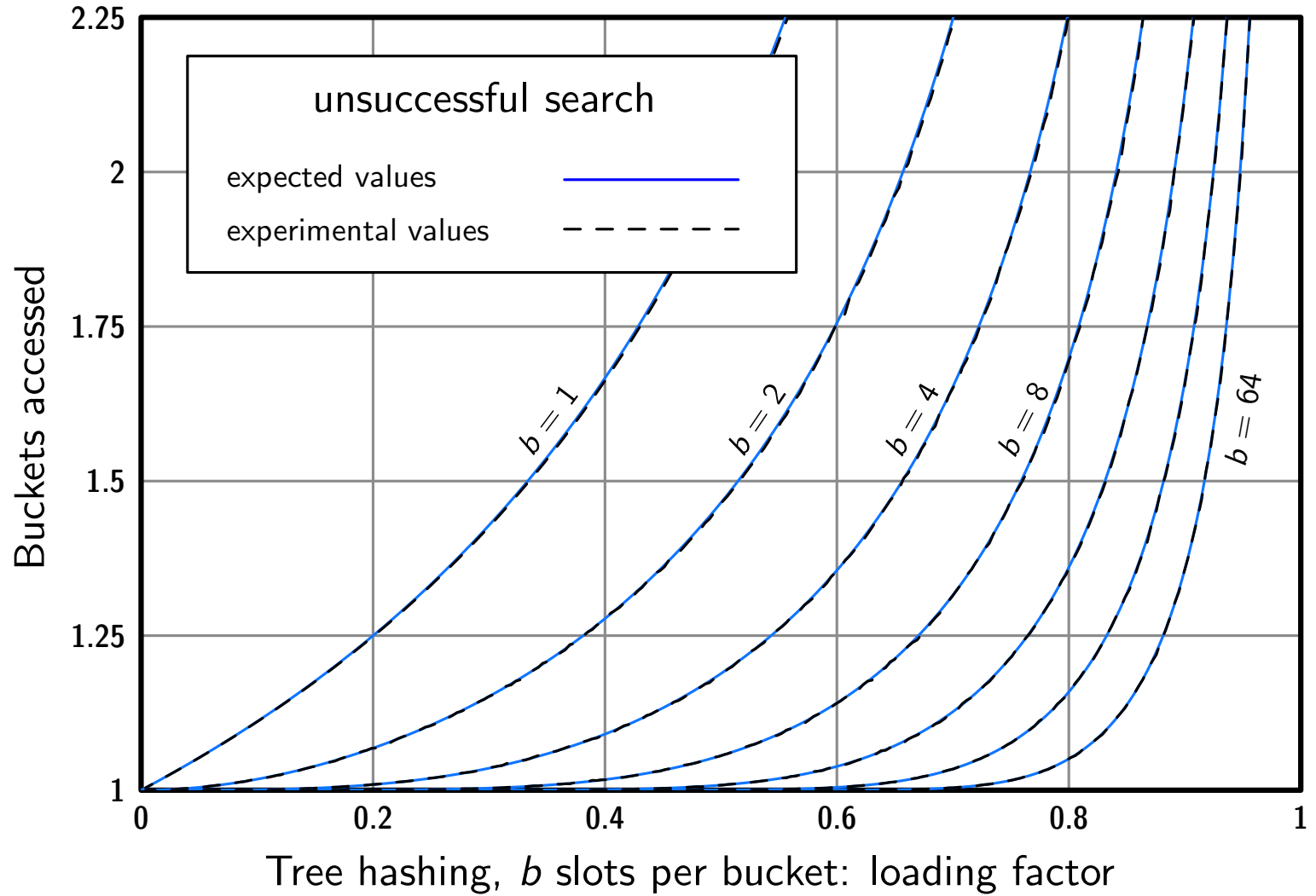
$$E[L_S] = \frac{1}{m} \sum_{k=0}^{m-1} \sum_{j \geq 0} \beta(k/(bn))^{((b+1)^j - 1)/b}$$

- Reasonably accurate for α no greater than 0.8;
7% error decreases with increasing b .

- **Unsuccessful search length** L_U

$$E[L_U] = \frac{1}{1 - \beta(\alpha)}$$

- Very accurate.





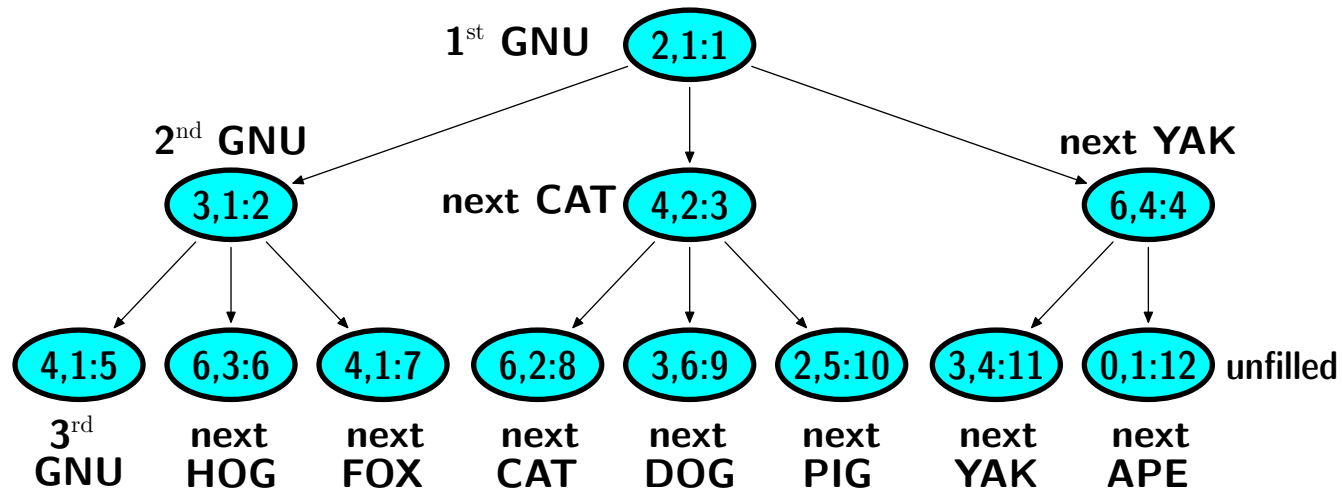
- Improved **successful search length** via Brent and then Gonnet and Munro.
- Let $p_i(\beta)$ be the probability that a record at a particular location will find at least the next i buckets probed to be filled; the sequence of i buckets is referred to as a chain. The quantity $\beta p_i(\beta)$ is the probability of finding such a chain at any table location.

Efficient Reordering of External Tables VIII



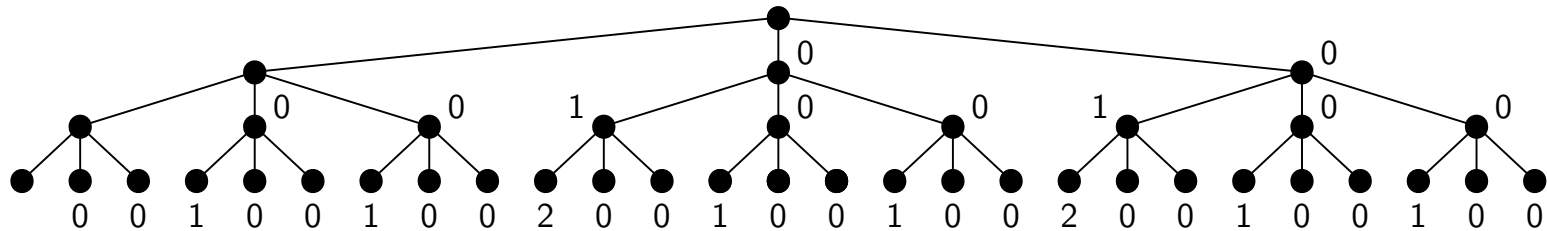
$$\frac{d(\beta p_v)}{d\alpha} = \beta^v + \frac{1}{1-\beta} \sum_{i=0}^{v-1} \beta^{v-i} (p_i(\beta) - p_{i+1}(\beta)) + \sum_{i=0}^{v-1} \beta^{v-i-1} Q_i(\beta)$$

$$p_v(0) = 0 \quad \text{for } v = 1, 2, 3, \dots \quad \& \quad p_0(\beta) = 1.$$





$Q_i(\beta)$ designates the sum of probabilities of all trees for which the tree traversal ends at a chain of length at least i .



$$Q_0 = \beta^2(1-p_1) \times$$

$$\underbrace{(1+p_1)}_{\text{second level}} + \underbrace{\beta(p_1^2+p_1^3+p_1^3p_2+p_1^4p_2+p_1^4p_2^2+p_1^5p_2^2)}_{\text{third level}} + \underbrace{\beta^2(p_1^6p_2^2+\dots)}_{\text{fourth level}}$$

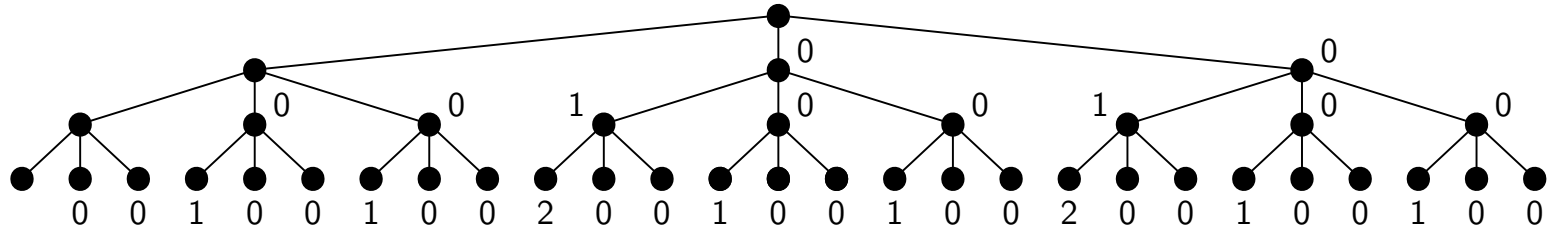
second level

third level

fourth level

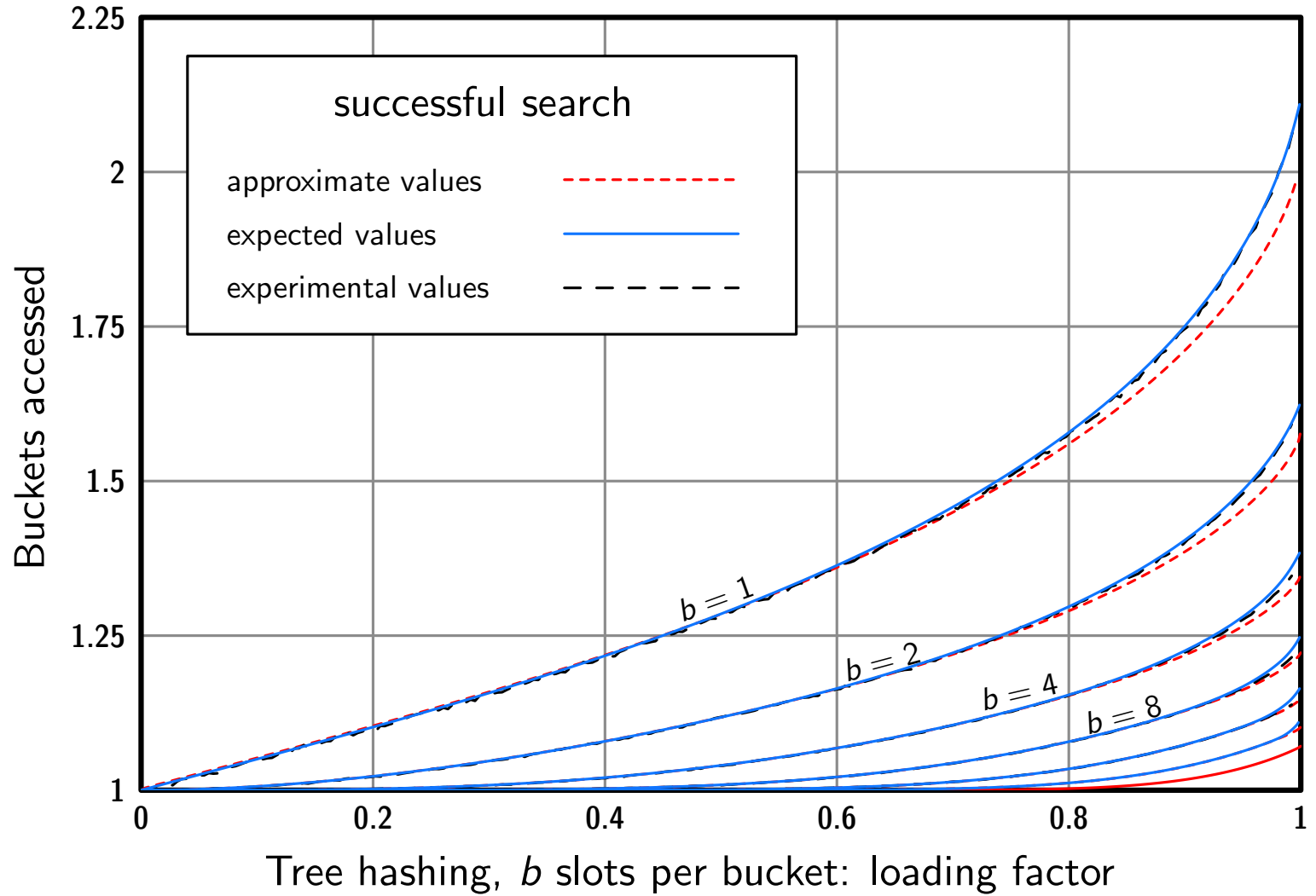
$$Q_1 = \beta^3(p_1-p_2) \times (p_1^3+p_1^4p_2 \dots)$$

Efficient Reordering of External Tables X



■ $IC(\alpha)$ is $1 + \beta + \beta^2 p_1^2 + \beta^3 p_1^6 p_2^2 + \beta^4 p_1^{18} p_2^6 p_3^2 + \dots$

■ $E[L_S] = \frac{1}{\alpha} \int_0^\alpha IC(t) dt$





- Conclusion and future work.

Tree hashing provides excellent expected access and insertion runtimes; without additional space.

Inclusion of passbits: extra space to reduce unsuccessful search lengths.

What is the best possible expected access runtime; it is known for capacity 1 buckets.