Problem addressed :

Traditional OS employ various types of scheduling algorithms which are inadequate. Priority based algorithms can be static, vary dynamically and sometimes priortities are arbitrarily changing. These systems provide crude control over scheduling as resource rights do not vary smoothly with priorities. Fair share schedulers are complex, they need timely usage updates and dynamic priority adjustments. Even microeconomic schedulers have assumptions and high overheads which limits them to relatively coarse control over long-running computations. Thus systems fail to provide responsive control over service rates.

Main idea :

The paper proposes lottery scheduling algorithm. Resource rights are represented by lottery tickets which are given to clients. A random lottery is hold for resource allocation, and the winning client gets the access to the resource. The algorithm is probabilistically fair. Number of lotteries won by a process has a binomial distribution. Also, number of tries before a process wins a lottery has a geometric distribution. The paper introduces currency abstraction, which is used to isolate trust boundaries as the inflation of currency is limited to a trust boundary. Compensation tickets approach is also introduced which make sures that CPU time is utilized totally.

Strengths :

* The algorithm provides modular resource management using the concepts of inflation, currency and compensation tickets. A process can thus avoid sharing its resource allocation policies with other processes.
* With help of ticket transfer concept, the problem of priority inversion is conveniently solved. A blocked process can transfer its ticket to other process for which it is waiting.
* The paper claim that the algorithm can be used to manage wide range of resources. I/O bandwidth, memory and locks along with databases can be managed. Space-shared resources such as memory can also be managed using a variant of this algorithm.

Weaknesses :

* Although the probability of a process getting a resource is non-zero, it is possible that a low-priority process can indefinitely wait for high priority processes to release the resource as there is increased chances of high-priority processes to win lotteries.
* The algorithm behaviour changes over time. The probabilistically fairness behaviour of the algorithm is achieved after a long run. Its results are not desirable for short time interval. The paper states that large ratios converge towards their allocated values over longer time intervals. Hence for short term, the algorithm is biased. The paper does not provide any tweaks or ideas to remove this short term biases.

Thoughts on paper :

Although the paper tried solving the issues of traditional scheduling algorithms, it failed to solved all the weaknesses mentioned in the paper like dynamically varying nature of priority. The paper talks about choosing process which has ticket to allocate the resources, but it is not specified how the ticket allocation process is done. The limitation of short term biases is also there and I am justified that the authors came up with the stride algorithm to counter it.