




J. H. Saltzer

Massachusetts Institute of Technology
Information Processing Center
Cambridge, Massachusetts 02139

To: Multics Administrative Distribution
From: T. H. VanVleck 
Date: September 21, 1971
Subj. : Load Control and Groups

This memorandum describes a proposed modification to the Multics administration scheme to allow for groupings of users who share some "gross" resources, such as guaranteed access to the system.

Introduction

The Multics system's resources are currently in demand during the first shift in excess of our capacity. During the period from July 7 to August 17, over five thousand attempted logins were refused because the system was full. Our hoped-for capacity increases during this period have either failed to materialize or have been used by users increasing their demands.

Although we may have some capacity improvement during the next few months due to a reworked paging mechanism and a new PL/I compiler, it is clear that the system is so successful that the time has come for the introduction of a rationing mechanism which attempts to give each user a fair chance to login.

Our current access-rationing mechanism allows us to specify a maximum number of simultaneous users for any particular project. This mechanism has the following deficiencies:

1. Projects are primarily file-access-control groupings. Some projects consist of only one user. Access to the system cannot be mapped easily onto file accessing privileges, and there are more projects than we can support simultaneous users. In some sense, projects are too "small" to use for system login rationing.
2. The current mechanism cannot support a guarantee to a project of a minimum number of users logged in.
3. If a user logs in early in the morning he may stay all day unless his project chooses to allow him to be bumped. Some groups take advantage of this policy by logging in several users first thing in the morning because they know they will be unable to do so later.

User complaints about the current mechanism come both from large projects who feel that they cannot get their share of the machine and from small projects whose users assert that they cannot ever get logged in. Furthermore, since "antisocial" behavior is profitable under the current policies, all users must act antisocially or lose out.

Groups

The solution proposed to the above problems is to introduce a higher-level administrative entity called a user group or group. The system will have a small number of groups, say no more than fifteen or twenty. Each ^{case?} project will be in one and only one group.

Groups will be strictly administrative-software concept -- that is, neither the directory hierarchy nor file access control lists will be modified to carry a group designator (at least initially).

Groups will function to allocate those system resources which are difficult to allocate at the project level. In general, all system resources and privileges should be subdividable or delegable, including the privilege of further subdivision or delegation. In particular, a group may subdivide its resources among its projects, and the projects may subdivide their share of the resources among individual users. Delegation will not be necessary, though: the central administration will operate any group not desiring to run itself, and a group may run any or all of its projects if no project administrator is available.

The rest of this discussion will focus on group control of login access to Multics. Control of other resources may indeed be attached to the group structure later, in a straightforward manner, but such additions to the group concept will be the subject of later design documents.

Gross Division of System Access

The system's user capacity will be divided among the various groups according to a fixed allocation made by the system administration. The number of load units thus subdivided will be the minimum reasonable capacity of the system (e. g. 41.0 units for the current M. I. T. configuration). Additional capacity provided by adding extra hardware will be managed to give all users an equal chance at it.

A group may establish its own strategy for allocation of the capacity delegated to it by choosing values of system-provided parameters. Users using the group's allocated capacity are called "primary" users (even if subject to bumping by other group members), and we will speak of the use of 1.0 load units of the group's capacity as a "primary line."

primary
load unit

When the system is underloaded, but a user's group has all of its primary lines in use, the user can still be logged in as a "standby" user (if his group permits this). In effect, the group "borrows" a line from some other group not using its allocation at the moment. A standby user may be promoted to primary status if enough users in his group log out or he may be bumped by a user from some other group entitled to primary line. This strategy attempts to let users log in as much as possible at the cost of having a standby user subject to bumping just after being logged in.

save how
units

Each group may control the following attributes for each user of the group:

1. Whether or not the user may use a "primary line"
2. Whether or not the user may be "standby"
3. The length of time the user is "protected" from bumping by other users in the group
4. Whether or not the user may bump other users in the group.
5. If a user has been bumped, the minimum time before this user may bump some other user on the group.

Need to
specify
the
bumping
algorithm

← If M₁ can't avoid bumping

Application to Current Situation

To give a practical example, here is a table showing a possible allocation for the current M. I. T. system:

<u>Group</u>	<u>Lines</u>
Overhead	5.0 (Sys Daemon, etc.)
System development	15.0 (Multics, MPM)
Other I. P. C.	0
Project MAC	4.0
Cambridge Project	3.0
All others	13.0

These numbers are just examples, of course. Actual values would be determined by the I. P. C. administration on the basis of revenue and administrative commitments. Note the non-Multics I. P. C. work is set to have zero primary lines. This allows usage but only when the system would otherwise be standby. Notice also that no attempt to break the "all others" group into one for the School of Science, one for Engineering, etc., as was done in CTSS, is proposed. Current I. P. C. policy says that a group is set up only for large "one-check" users who generate enough revenue and who can provide effective group supervision.

Edit-Only Service

When the system is full, a user unable to log in is completely helpless. It is as if we locked the keypunch room whenever the batch machine's queues were filled. If a sufficiently low-cost editor were available, it would be nice to allow a couple of users on after we had reached our maximum load, restricted to editing only. These users would have a governor set to prevent them from using more than a tiny fraction of the system's capacity.

Looked at another way, this is an improvement on the "queue for login" idea suggested by several users. If the user is connected to the system but we cannot handle the load we expect from him, it would be nice to allow him to edit his files and submit absentee requests.

We have almost all the part necessary for such a mechanism now. Our

editors are believed to be relatively low-cost (this should be studied though) and the governor mechanism is operable. All that is required is some code in the answering service to force a user into edit-only status when the system is full and to promote him when he has a chance to log in.

Scheduled Service

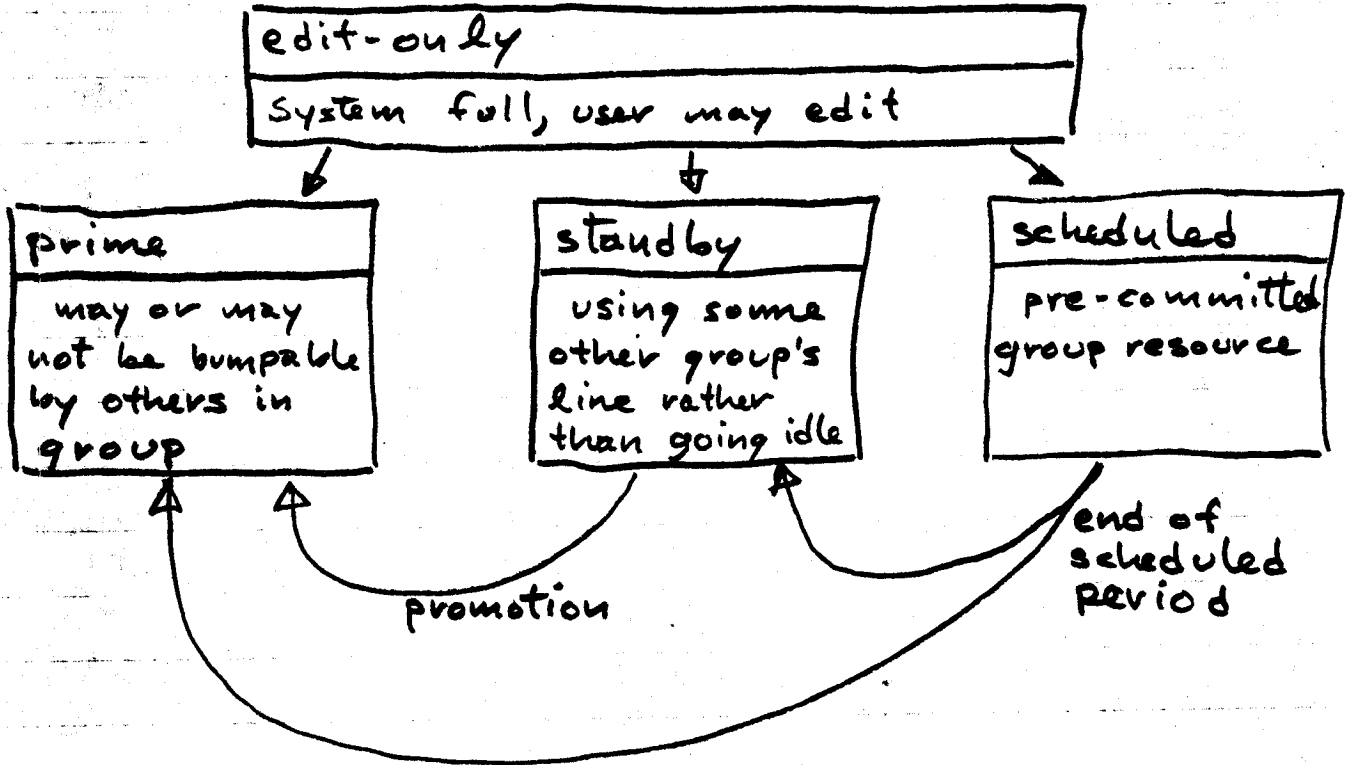
Many applications of Multics, especially administrative and "production" use, would rather not have any competition for or uncertainty about getting logged in. For these groups a "scheduled service" concept has been invented. It will also serve to help demonstrations of the system or its subsystems, and to satisfy the demands of users with unusual time constraints.

A group may withdraw one or more primary lines from its regular pool and devote them to scheduled service. A scheduled line will be reserved in advance for a particular user (or project) for given blocks of time, half an hour long. When a user attempts to log in he is checked to see if he is scheduled and if so he may always log in.

If a scheduled line is not being used by the user who reserves it, it is available to standby users.

When a scheduled user's time is up, he is put on primary or standby status depending on whether or not his group's unscheduled capacity is consumed.

User States



Other Remarks

The system's capacity must not be over-allocated. Otherwise, we can default on scheduled service or on number of primary users.

Scheduled service is cancelled when the system is down. Coding a "readjustment" algorithm is probably just too complicated.

Per-session, per-day, and per-month limits on connect time should also be provided.

Balanced Service

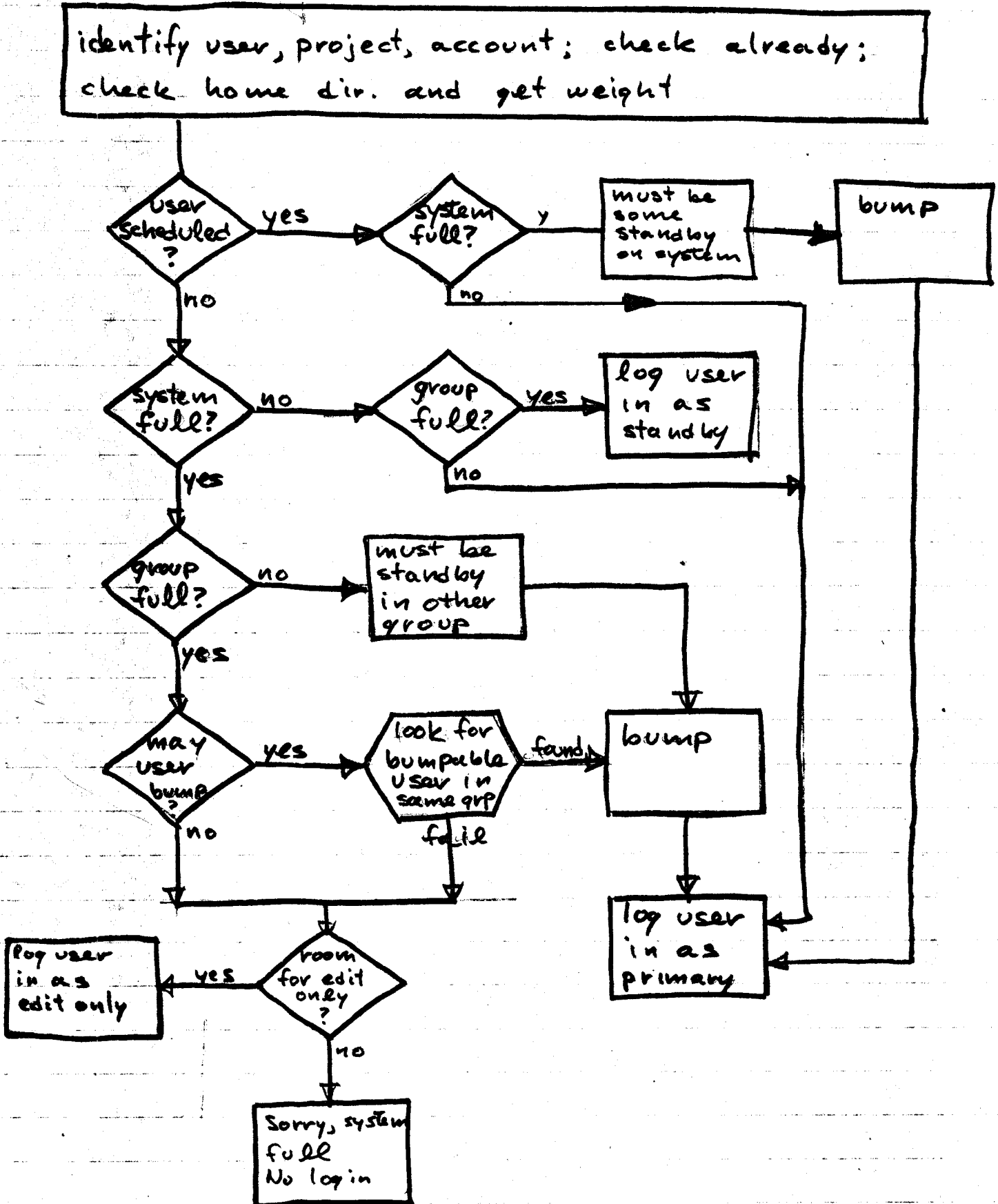
This proposal sounds complicated but looks useful. It is an attempt to provide a self-running allocation system within a group to decide whether or not to bump one group user in favor of another. It is an attempt to provide a system with "memory," such that a user who is allocated a certain share of the group's resources over a month will have a better chance of getting logged in if he has not logged in recently.

Each user will be allotted a certain number of connect hours. Usage (perhaps first-shift only) decreases this balance. The balance is replenished daily by an increment, up to some maximum.

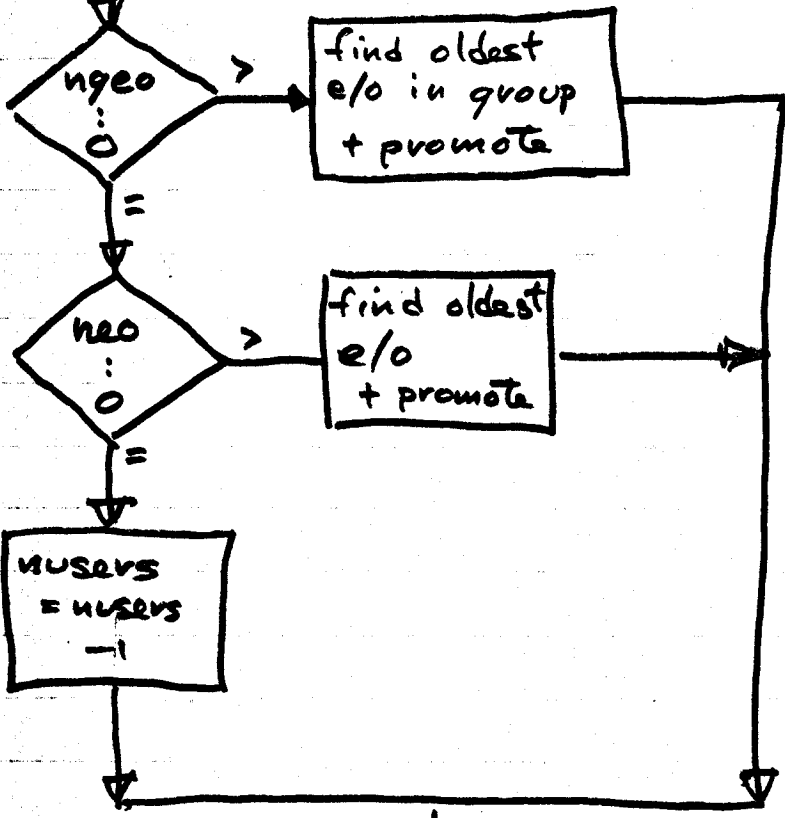
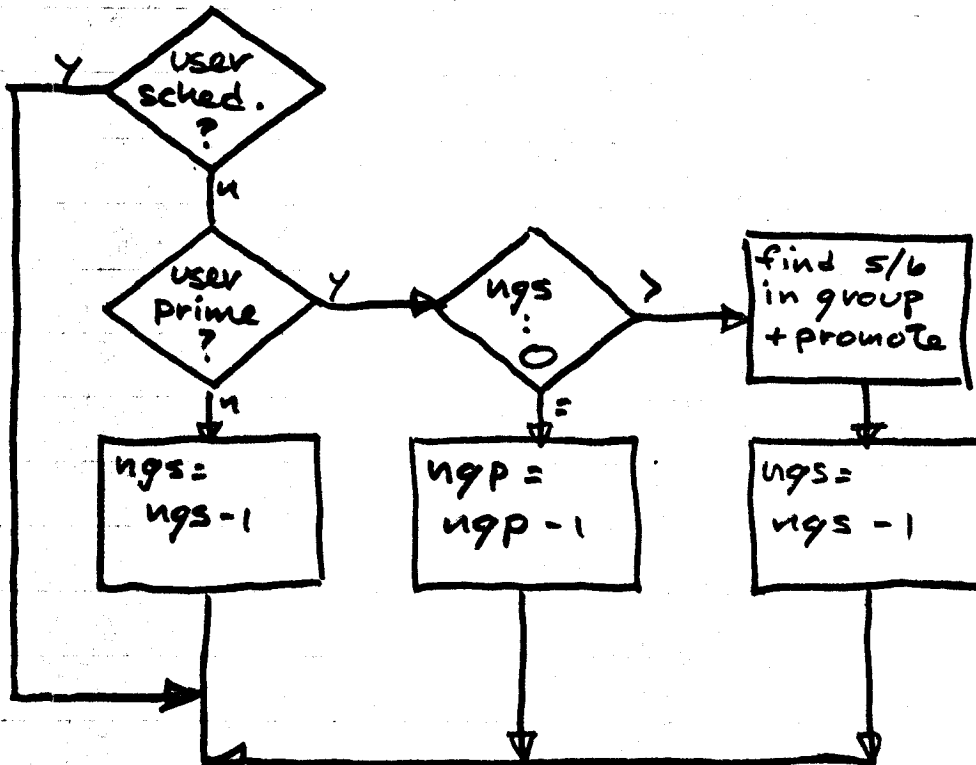
When the system is looking to bump one user on a group for another, a user will be able to bump users with a lower balance only and will bump the user with the least balance.

Several users have expressed interest in this scheme. The only real problem I see is explaining it to users, especially if the scheme needs adjustment. Perhaps it can be released to a "guinea pig" group first.

User requests login.



Promotion when user logs out.



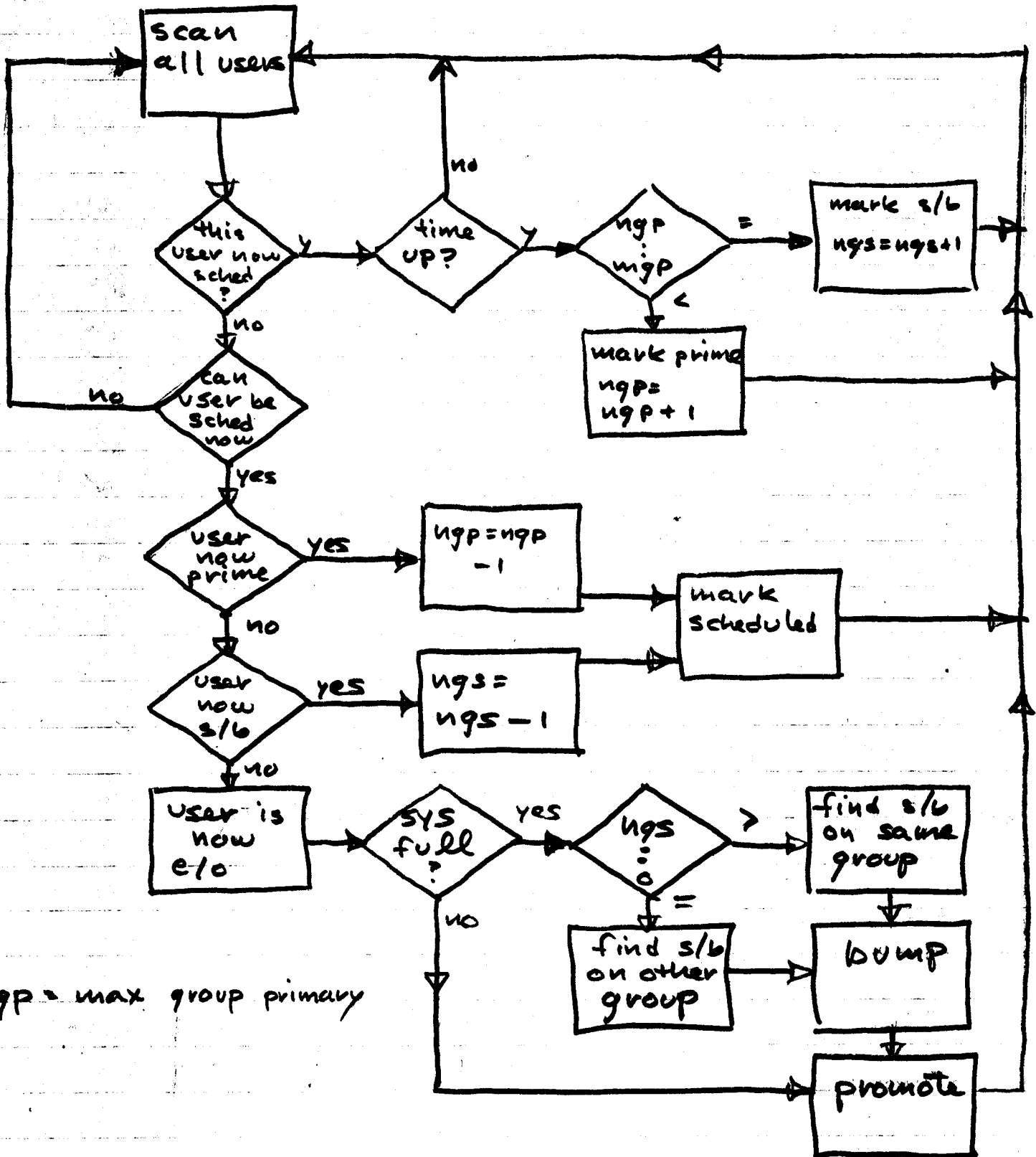
ngs = no. of group standbys

ngp = no. of group users primary

ngeo = no. of group edit-only



Routine called when schedule period ends to adjust status of scheduled users.



ngp = max group primary

Example

Management of the "All Others" group.

1. Bump grace for all users will be two hours.
2. All users may use prime.
3. All users may be standby.
4. On line for scheduled service, sign up basis, with rules
 - a. Only two hours at a crack
 - b. Downtime cancels schedule
 - c. Only two hours/week/project
 - d. Register a week in advance
5. After being bumped user may not bump for one hour.
6. Balanced service is not operating. (We could do it by giving the same to all users.)

Implementation

A new system table will be required. It will contain the line allocations, limits, and current status for each group. Conversion and installation programs will be required.

The lg-ctl-module will require a great deal of modification, as shown in the flow charts.

"Who" will need modification to list edit-only users, line status (primary or standby), group affiliation, and line totals.

The system administration table will need a new field designating the group for each project and, perhaps, other changes.

Minor changes will be necessary to support edit-only users.

Balanced service will need some additional work to support the incrementing balances.

Both edit-only and balanced service should be planned for from the beginning but not actually implemented on the first pass.