NAME

ldb – Play backgammon by e–mail

SYNOPSIS

ldb [options] [input files...]

DESCRIPTION

Ldb allows two people to play backgammon over a network using electronic mail. It runs on character–oriented terminals, or emulators thereof, using the *curses* screen package. It will run on most UNIX dialects, as well as VAX–C under VMS 5.0 and above. *Ldb* will display a picture of the board, prompt the user to enter a move, check the move for legality, and send the move to the opponent. When it arrives, it will be displayed on the opponent's screen, after which the opponent will be prompted for a move to be sent back.

FEATURES

Multiple Games

Ldb can keep track of an arbitrary number of games simultaneously, and will display them one at a time for you to play. You can exit from *ldb* at any time; when you run ldb again, it will pick up your games where you left off.

Move Checking

Both your moves and your opponent's moves are checked for legality. This includes:

- Landing on an occupied point.
- Bearing off before your inner table is full.
- Trying to move pieces that are on the board when you have a piece on the bar.
- Trying to bear off a piece when there is a piece on a higher point, and the roll is not being used exactly.
- Trying to leave a usable move unused.
- Trying to leave a move unused when it is usable and has a higher roll than the move that is used.
- Trying to double when you doubled last.
- Trying to double when either player is within 1 point of winning the match, and the Crawford rule is in effect.

Message Reliability

Ldb detects (and rejects) duplicate messages, and can regenerate lost messages.

Remote Start

Ldb can start a game between two players on remote hosts. This is used to set up "game starter" hosts, where people interested in finding new opponents are matched by some criteria (such as ability, or by a tournament schedule), and have their games started automatically. For a current listing of game starter hosts, see the README file.

Game Value

Ldb detects the end of the game, gammons, and backgammons, and notifies both players of the game value when the game is over. The Jacoby rule may be enabled, as may European scoring, to calculate the game value as desired.

Online Help Screen

Ldb has an online, context sensitive help facility that lists the available commands for the current screen.

Match Play

Ldb can support matches that play to any desired point total, and will continue to start games until one player reaches that total.

OPTIONS

-read Mail is read, and the games are updated and saved. The user is not prompted for his moves.

-play Any games that are waiting for local input are displayed for the user to process. No mail is read.

-color xy

The colors for any games started are set to x and y. The first color is played by the local user, and the other is played by the opponent. Legal color characters are any upper or lower case letters. The default is "–color rw". This argument is positional, and only affects games started after it has been scanned.

-direction up/down

The direction of play for any games started by the local user is set to the specified value. The default is "–direction up". This argument is positional, and only affects games started after it has been scanned.

-myaddr addr

Your mail address is set to *addr* for games started after the *–myaddr* option is scanned. This overrides the address specified in *.ldbrc.*

-start user

A game is started with the specified user. The *user* argument may be any e–mail address that is valid for your host. Some e–mail addresses will need to be quoted if they contain shell metacharacters. The *user* argument may also be an *alias* for an opponent you have played before. Aliases are defined in the people file, and are initialized to the first name of the opponent, with all upper-case letters changed to lower.

-remotestart user1 user2

A game is started between user1 and user2. The local host sends a remote start message to user1 instructing it to start a game with user2. The local host does not participate thereafter in the game. For the purposes of the –color and –direction options, user1 is considered the local user.

-jacoby The Jacoby rule is enabled for all games started after the *–jacoby* argument is scanned. This rule specifies that gammons and backgammons count as single games if neither player doubled during the game. This option may only be enabled by the user starting the game, and may not be disabled by the other player. If the opponent's version of ldb does not support the Jacoby rule, it is disabled and the game proceeds as if it were never enabled.

-crawford

The Crawford rule is enabled for all games started after the *–crawford* argument is scanned. This rule specifies that neither player may double if either player is within 1 point of winning the match. If match play is not enabled, this argument has no meaning. This option may only be enabled by the user starting the game, and may not be disabled by the other player. If the opponent's version of ldb does not support the Crawford rule, it is disabled and the game proceeds as if it were never enabled.

-european

The European rule is enabled for all games started after the *–european* argument is scanned. This rule specifies that backgammons count as double, rather than triple, the score shown on the cube. This option may only be enabled by the user starting the game, and may not be disabled by the other player. If the opponent's version of ldb does not support the European rule, it is disabled and the game proceeds as if it were never enabled.

-permanent

Any games started after the *–european* argument is scanned are marked permanent. Permanent games are automatically restarted when they end. This is for people you play with constantly, and it saves you the trouble of restarting the game when it is over. This option may only be enabled by the user starting the game, and may not be disabled by the other player. If the opponent's version of ldb does not support permanent games, the permanent flag is disabled and the game proceeds as if it were never enabled.

-match points

This option enables match play and specifies the number of points the match is to play to. Games that are part of a match have additional information displayed above the board to show the current match score and the number of points being played to. When a game that is part of a match ends, and the specified number of points has not been reached, the game is restarted. This option may only be enabled by the user starting the game, and may not be disabled by the other player. If the opponent's version of ldb does not support match play, the game proceeds as if it were a normal game.

-broadcast file

The specified file is mailed to all current opponents. This is useful for announcing vacation absences, etc.

The default mode for the –read and –play options is to do both. This is also what happens if both –read and –play are given. Any arguments not beginning with – are treated as input files and are searched for incoming moves from opponents. It is not an error for any of these files to not exist.

-control

This option puts ldb into a special mode that does not allow games to be played, but instead displays them one at a time and allows certain "overhead" functions to be performed. These are described fully in the section titled "CONTROLLING A GAME".

-reconstruct file

This option allows your game file to be reconstructed if you accidentally clobber it. It does this by extracting the information from your opponent's data file and "reversing" it so you can use it. To use the reconstruct option, ask all of your opponents to send you their .ldbdata files. Extract these from your mail into files on your system, strip the mail headers, and feed them one at a time into ldb. For each game found listing you as the opponent, you will be prompted to enter the opponent's e-mail address or alias (and his name if he is not in your people file), and the game will be added to your game file.

The reconstruct function cannot recover games that were in the process of starting (i.e. in their initial roll stage), or that have already completed (why bother?). Your opponent should read any mail files you sent before your data file got clobbered, and should make his move and send it before sending you his data file. You will not need to process the move he sent, though, since all that data is in his file anyway. It won't hurt if you do, ldb will just ignore it.

-help This prints a one-line description of each option, in case you forget later on.

SCREEN

The screen displays the following items of interest:

Opponent
The e–mail address and personal name of the opponent are displayed on the top line of the screen. This can help identify the game if you have many games in progress at once.

Board Label
Ldb displays one of three possible board images at a time; the name of the board currently displayed is shown on the second line of the screen, near the left. The significance of these three boards is explained below.

Flags This area displays any special rules in effect for the game. Each special rule causes one character to be displayed. These are:

J	Jacoby rule
C	Crawford rule
E	European rule
P	Permanent game

Match Score
The current match score is displayed next to the board label, as is the number of points the match plays to. If the game being displayed is not part of a match, this area is blanked out.

Pip Count
Near the right of the screen on the second line is the pip count for the board being displayed.

Board The board is displayed near the left side of the screen, and takes up most of the screen area. It consists of an outline, drawn with ASCII characters, and 28 "points" where pieces can reside. Of these, 24 are the normal points making up the main body of the board (numbered 1–24), 2 are the "BAR" points, located between the left and right halves of the main board, and 2 are the "OFF" points (where pieces go when they are borne off), located to the left of the board. Pieces are represented by upper or lower case letters. To help you remember which direction you are playing, there are arrows drawn at the four corners of the board indicating your direction of play.

Doubling Cube
The current value of the doubling cube is displayed in reverse video on the left side of the screen. The cube will appear next to the home board of the owner of the cube, or in the middle if neither player has doubled. For example, if the player whose home board is at the bottom of the screen doubled last, the cube will appear beside the top of the board.

Move Blocks
The move blocks are located on the right of the screen. There are two move blocks; one for you, and one for your opponent. You will use your move block to enter your moves,

and you will see your opponent's moves displayed in his move block. A move block consists of a header, which identifies the owner of the move block, and two columns of either two or four lines. Each line in a move block is one roll, and shows the value of the roll (in the range 1–6) and how that roll has been used. If doubles were rolled, the third and fourth lines will also be used to display the two additional rolls that may be used.

Command Box

The bottom right corner of the screen contains a small box that displays the commands that are legal at any given time. You can execute a command by typing the first letter; you should not hit return afterwards. There may be legal commands other than those shown in the command box, depending on the state the game is in. Press "h" or "?" for a complete listing of legal commands.

Message Area

The bottom 6 lines of the screen are reserved for messages of various types. The bottom line is used for error and informative messages that occur as a game is played. The top line in this area displays messages pertaining to the status of a game, such as blots hit by an opponent's move. The remaining 4 lines are used to enclose a message to your opponent with your move, and to display the one your opponent sent to you. These messages may be up to 2 lines of 56 characters.

STARTUP FILE

The first time you run *ldb,* it will ask you if you want to create a *.ldbrc* file. You should answer "yes", unless you have a reason not to. If you want the ldb startup file to be named something other than *.ldbrc,* you should set the environment variable LDBRC to the name you want before running ldb for the first time. If you do this, the LDBRC variable must be set every time you run ldb. *Ldb* moves to the directory in the HOME environment variable, if it is set, before accessing any files. This makes all files (including .ldbrc) relative to $HOME.

The *.ldbrc* is organized as a sequence of name/value pairs, one per line, in the format *name=value* . There should be no spaces before *name* or before or after the equals sign. Comments are indicated by a pound sign (#) in column 1, and extend to the end of the line. The following names are defined:

myname

This line is where you enter your (human) name. It is sent to your opponent when a game is started, to make things a little more personal. You can put anything you like here, of course, but I highly recommend you just stick to your name. The *myname* line is mandatory; if you don't want your name sent to people, you can just put your email address instead, but you must have something.

myaddr This line defines your e–mail address. This is sent to your opponent when you start a game so they know how to send mail back to you. For people with multiple mail paths, put the one here you use the most (e.g. the Internet domain–style path), and for games that need to use a different address, use the *–myaddr* command line argument to override the path in *.ldbrc. Note that the first time you start a game* with someone, the address you use is saved in your people file. Subsequent games you start with this person automatically use this same address, so the *–myaddr* option will be unnecessary. The *myaddr* line is mandatory.

peoplefile

This line sets the file name that is used to store information about your opponents. In addition to storing your opponents' names, addresses, and aliases, it records the number of games, points, gammons, backgammons, and matches you have won or lost to that opponent. It also stores something called the *fence,* which is the start time of the most recently finished game. This allows start messages from old games to be discarded. The peoplefile line is optional, and if it is not given, the people file defaults to *.ldbpeople.*

gamefile This line sets the file name that is used to store games in progress. It is optional, and if it is not given, the game file defaults to *.ldbdata.* My advice, if you care, is that you should avoid using a file name containing the string *games* or any other giveaway string that would let your boss, sysadmin, or other local Nazi–type know that you're using a very expensive piece of high–tech equipment to goof off, when you could do that with nothing more exotic than a rubber band and some spitballs.

backupfile

This is the file your old *gamefile* will be moved to just before a new one is created. It defaults to *.oldldbdata.*

mailfile This line is where you tell ldb how to find the mail messages your opponents have sent. There are several ways to do this. The simplest one is to just put some file name here (the default is ldb.rcv) and, when you receive mail containing an ldb move, just save it to this file. When you are done reading your mail, run ldb, and it will scan this file and extract all your incoming moves. If your mail program is like most, it will append messages if the file you are saving to already exists, which is just what you want. *Ldb* will silently ignore moves that it has already seen for games in progress, so you don't have to clean them out of your mailfile if you don't want to. Note however that if anything happens to your people file, and you have old start messages in your input file, ldb could start a spurious game (i.e. one your opponent's ldb knows nothing about). Also, if your input file gets really long, ldb will take a while to start up. So the best course is to delete all messages after ldb has read them.

For UNIX users, a much easier way is to set the mailfile to point to your actual mail file (e.g. /usr/spool/mail/$USER), and let ldb scan your mail directly. This will only work if your system keeps mail in ASCII format, but many do. Ldb will not delete these messages from your mail so you will have to.

The third option is to set the mailfile to a pattern instead of a filename. Ldb will scan all matching files for moves. This is good for mailers that don't append on save, and don't store in ascii format, such as on VMS. Note that on VMS, if no version number is defined, mailfile defaults to scanning all versions. VMS users should read the section entitled "VMS NOTES" for more information on extracting mail.

delmail This option causes any mail files scanned by ldb to be removed after they have been scanned. This is a hideously dangerous option, so I recommend you try it in a friend's account first. It will delete ANY file that matches the pattern specified in the mailfile line, with the exception that, under UNIX, it will refuse to delete any file starting with "/". Use it with great care. This option defaults to no; to enable it, set it to "yes".

lockfile This line sets the name of the ldb lock file. It defaults to ".ldb_lock", and there's no real reason to change it unless by some chance it conflicts with a file name you might use. When ldb starts, it checks to see if this file exists; if it does, ldb exits. Otherwise, it creates it and goes on. This prevents more than one ldb from running in the same account at the same time, and thereby garbling the data files.

sendcmd

This tells ldb how to send a file with your particular mail command. This string is passed to the system, after making the following substitutions:

$a is substituted with the address to mail to.

$f is substituted with the file containing the message.

$s is substituted with the subject of the message.

Most mail systems will work fine with the default, which is

mail –s '$s' $a < $f

or, on VMS,

MAIL/SUBJECT="$s" $f IN%"""<$a>"""

Note that the $ substitutions are made by ldb, not by a shell, and thus are not subject to any quoting. If you need a $ in the string, use $$. VMS users may need to change the name of the SMTP mailer; for example, to send mail using Wollongong TCP/IP, you would use the string

MAIL/SUBJECT="$s" $f WINS%"""<$a>"""

tempfile This line specifies a temporary file that ldb can use to compose mail messages. The file will be created in your home directory, thus avoiding file name conflicts in /tmp. It defaults to ldb.tmp.

colors This is a string of two upper or lower case letters, and defines your preferred colors for any games you start. The first character in the string is the color you will play, and the second is the one your opponent will play. It defaults to "rw", and is overridden by the *–color* command line argument.

direction
This is either "up" or "down", and defines your preferred direction of play for games you start. The direction of play only refers to whether you play toward increasing or decreasing numbered points. Although by default upward play goes toward the top of the screen, this can be reversed (for your screen only) by the "Invert" command in the Control menu, accessible via the *–control* command line argument. The default for this line is "up", and is overridden by the *–direction* command line argument.

initialboard
As was alluded to earlier, your screen shows one of three possible board images at any time. Although the purpose for this will be explained later, this line allows you to select which of these three boards should be displayed initially. The legal values are "before", "after", and "current". The default is "current".

autoroll If you were the last to double, you have no option but to roll the dice when it is your turn. If autoroll is enabled, ldb will roll the dice for you automatically in this case. For those that prefer consistency, autoroll may be disabled, although attempting to double when you doubled last will cause an error message. Possible values for autoroll are "yes" and "no", with "yes" being the default.

automove
Each time you roll the dice, ldb determines how many legal ways there are for you to use your rolls. In the relatively unusual case where there is only one way for you to use your rolls, ldb can automatically apply this move to your board before displaying it. Automove is disabled by default because quite frankly it's pretty annoying. Possible values for automove are "yes" and "no", with "no" being the default.

autodouble
One of the official rules of backgammon allows for automatically doubling the game value when the opening roll is a tie. The rules go on to state that automatic doubling is only to be used when the players agree beforehand on its use and limitations. The method ldb uses to implement this somewhat vague rule is to allow each player to specify a maximum

number of autodoubles they are willing to accept in *.ldbrc,* and for the maximum number of autodoubles applied to any game to be the lesser of the two players' specified limits. For example, if you set *autodouble=4* in your .ldbrc, no game you play will ever be autodoubled more than 4 times, although the number of tied opening rolls is obviously not limited. However, if you were to start a game with someone who had set *autodouble=2* , that game would be limited to at most 2 autodoubles. If you don't want to play automatic doubles, set autodouble=0, which is also the default.

superkey
: If you live in fear of roving supervisors (or other fun-impaired individuals) running across you playing an innocent game of backgammon, you may want to enable the superkey function. Just pick a key on the keyboard you never use in ldb, such as the backquote, and put a line in *.ldbrc* setting superkey to the *decimal ASCII code* of the key you want; for example, to use the backquote key, use "superkey=96". Only single–character keys are allowed, and the key you select will not be usable for normal purposes within ldb. Make it one you can hit in a hurry.

supercmd
: If you'd rather have a program of your choice come up instead of the shell when the superkey is pressed, put it on this line. It will be passed unchanged to a shell via the system() function. VMS users should read the note in the section titled "VMS NOTES" on supercmd functionality.

checkpoint
: This option keeps the gamefile up to date by writing it out whenever the dice are rolled or a message is sent. This prevents game information from being lost if ldb crashes, your machine goes down, or what have you. This option is enabled by default, and I highly recommend you leave it that way. If you just can't stand the thought of a few extra writes to disk, you can disable it with "checkpoint=no". On your own head be it.

debug
: This option enables various debug messages. It is a bitmap, with each set bit enabling debug messages from a different part of ldb. Most people will want to leave it at 0, which is the default. If you want debug messages, look for the definitions of DB_* in ldb.h.

PEOPLE FILE

Ldb maintains a list of all of the people you have ever played. This file is used for several purposes:

1. It stores the number of games, points, gammons, backgammons, and matches you have won and lost to that person.

2. It stores the address for that person, and the address that person should use to get to you. It also allows you to assign an alias to that person, and to use that alias when starting a game instead of the full e–mail address.

3. It stores information that allows ldb to reject start messages from defunct games.

Normally, you should never have to edit the people file, but it is described here in case you need to for some reason. It is in the same overall format as .ldbrc (name/value pairs).

addr
: This is the address of the opponent. This is what uniquely defines the record, in case you are playing two people with the same name.

name
: This is the name of the opponent.

alias — This is a "nickname" you can assign to an opponent. When a new opponent is found, the alias is initialized to the first word of the opponent's name, with all upper case letters changed to lower. You can change the alias to any desired string with a text editor (just make sure ldb isn't running at the time, or your changes will be lost).

myaddr — This is the address that this opponent should use to mail to you. This saves you from having to specify the -myaddr argument when starting a game, and is especially important with permanent games, since these games are started automatically without giving you a chance to change your address. If you only have one e–mail path, you can ignore this. This field is initialized to the value of the *–myaddr* option at the time the first game with this opponent is started, or to the value of myaddr from .ldbrc if no *–myaddr* option was used.

equiv — The presence of this field flags this as an "equiv" record. Equiv records allow one opponent to have multiple e–mail addresses, but to cause all of them to refer to one people record. Equiv records have no valid fields except *addr* and *equiv,* and must be preceded by the record they refer to in the people file.

fence — The fence is defined as the start time of the most recently started game that has finished. If that makes no sense, don't worry, you should never have to mess with it. It is used to discard old start messages that you forgot to clean out of your mail.

score — This is an array of 10 numbers that represent, in order:

- Games won
- Games lost
- Points won
- Points lost
- Gammons won
- Gammons lost
- Backgammons won
- Backgammons lost
- Matches won
- Matches lost

This information is stored purely for whatever purposes you can think of.

end — This line is required to separate people records. If you hand enter a record, you must be sure to put an end marker or your record will merge with the next one, which is not what you want.

STARTING A GAME

To start a game, you must use the *–start* command line argument, along with the e–mail address of the person you wish to play with. For example,

```
ldb –start mybuddy@back.gammon.com
```

If you want to play 'w', and make your opponent play 'b', use the command

```
ldb –color wb –start black@blah.blah.blah
```

See the OPTIONS section for more details on the available command line arguments. *Ldb* will not allow you to play with yourself, but that's ok because you'd probably lose.

When you start a game, ldb sends your e–mail address to your opponent, along with your opening roll. When your opponent receives your message, his ldb will roll his opening roll, and decide who won. If you won, his ldb will send your ldb a message to that effect, and the board will come up on your screen with that roll in your move block. If he won, he will use the opening roll and his ldb will send your ldb a message telling you how he moved. Either way, the game is off and running.

If, by unfortunate chance, the opening roll is a tie, your opponent's ldb will send your ldb a message to that effect. This will cause a new opening roll sequence as described above. A message will be printed telling you what's going on; just keep feeding the incoming messages to ldb, and it will take care of everything.

PLAYING A GAME

When it is your turn, the game will be displayed on your screen, and you will be faced with three options: Roll, Double, or Concede. Unless you are very cocky, I would not suggest doubling at this point. You could always concede, but I really think that would be selling yourself short. So you should choose ”Roll”. Just press ”r”, and watch your roll appear on your move block (along the right side of the screen, under the word ”You”). You should see two numbers appear (four if you rolled doubles), and the word UNUSED beside them. Decide which of the rolls you want to use first, and press the number of that roll. For example, say you rolled 1 6, and you want to move the obvious 8–7, 13–7 to make your bar point. Just hit ”6”. Notice the cursor jump up to your move block, beside the ”6” roll? It is waiting for you to enter the point you want to move from. Press ”1” ”3” (don't hit return). You should see a piece move from 13 to 7, and this fact will be reflected in your move block as well. Now, press ”1”. Again the cursor jumps to the move block, and you type ”8”. The second entry in the move block reflects this move, as does the board. Happy? If not, press ”r”; this will reset your rolls to UNUSED and put the board back to how it was before you moved, and then you can enter some different moves. Otherwise, hit ”s” (not ”q”!) to send the move. The cursor will jump down to the message area, where you get a chance to enclose a message to your opponent, perhaps to gloat over the great move you just made. You only get 2 lines of 56 characters, so keep it short. When you are done, press <return>. Your move is on its way!

There are three shortcuts you can use after you have entered your first move:

1. You can press <return>, which will ”repeat” your last move. That is, your next remaining roll will move from the same point as your last one moved from. This is handy when using doubles to move more than one piece from the same point.

2. You can press <space>, which will ”continue” your last move. That is, your next remaining roll will move from the point your last move ended on.

3. You can press ”p”, which will attempt to make the point your last move ended on. In our 1 6 example above, you could have just hit ”p” after you moved 13–7, and ldb would have figured out that to make the 7 point with the remaining roll of 1, it would have to move a piece from the 8 point.

In addition, there is one more shortcut that may be used with any roll. If you want to use a roll to bear off a piece, you can just hit ”o”. Ldb will locate the point (if any) from which pieces may be legally borne off by the next unused roll, and make that move automatically. You can also select a roll, then hit ”o”.

If you use your moves, hit ”Send”, and start to enter your message, then realize you made a mistake, you can press ESC to get out of send and back into the game, where you can use ”Reset” and re–enter your moves.

DOUBLING

If you decide to double, just choose ”Double” instead of ”Roll”. You will get to enclose a message, just like when you send a move, then your message gets sent. Once you have doubled, you cannot double again until your opponent does so, so ldb automatically selects "Roll" and sends you to the next menu. If you don't like this, you can disable it by setting *autoroll=no* in .ldbrc.

If your opponent doubles, you will see a message just under the board to that effect, and a menu will come up allowing you to Accept or Decline the double. Declining the double means you lose the game for the current game value. Accepting means that the game continues, but its value is doubled. The current game value is shown to the left of the board. After you accept, you should see this display move next to your home board, indicating that you own the doubling cube.

If the Crawford rule is in effect, and the either player is within 1 point of winning the match, doubling is illegal, and will result in an error message.

BOARDS

Ldb maintains three board images at all times. These are:

1 The board before your opponent's last move. It is for display purposes only, so you can see your opponent's move "before and after". The pieces your opponent is going to move are highlighted.

2 The board after your opponent's last move. It shows the condition of the board after your opponent's last move, with the pieces he moved highlighted.

3 The current board. This is the "real" board that shows the current state of the game. Nothing is highlighted.

By default, ldb displays the Current board at all times. If you want to see the other boards, use the Board command (provided it is in the current menu). This will cycle through the three boards in the order they are listed above. The second line of the screen tells you which board is displayed (near the left). When you roll, ldb automatically switches to the current board if it is not already up. You can set which of these three boards is initially displayed via the *initialboard* setting in .ldbrc.

ENTERING COMMENTS

Whenever you send a move to your opponent, you are given the opportunity to enclose a two line comment. While entering your comment, certain keys have special meaning. These are:

BS/DEL
: The backspace and delete (or rubout) keys both erase the last character typed. You can backspace from one line to the previous one.

^W/^B
: The ^W and ^B (control-W and control-B) keys both erase the last word typed. Words are delimited by spaces.

^X/^U
: The ^X and ^U (control-X and control-U) keys erase the current line and move to the end of the previous line, or to the beginning of the first line if used on the first line.

RETURN
: If you press return on an empty line, or on the last line, the comment is finished. Otherwise, the cursor moves to the next line.

ESC
: The escape key aborts the operation (e.g. Send), restores the previous comment, and returns to the previous menu. This allows you to change your mind and back out of an operation.

All other keys are entered into the comment. When you reach the end of a line, ldb word-wraps to the next line, or beeps if you are at the end of the last line. Note that the superkey is still active while entering comments, and cannot be entered into a comment.

CONTROLLING A GAME

The *–control* option permits you to control your games in certain ways. Each of your current games are displayed, one at a time, just as they are when you are playing, allowing you to perform any of the functions on the control menu. The control menu has the following commands:

Invert
: This allows you to invert the board for this game. The board will invert immediately on your screen, so you can see if you like it. Pressing *Invert* again will put it back to normal. Inverting the board does not affect the point numbering or the color you play, it merely draws the board with the low-numbered points at the top of the screen rather than the bottom. You may find it easier to make all of your games play a certain direction.

Resend
: From time to time, messages may be lost due to faulty mailers or human error. If your last message gets lost, you can regenerate it with the *Resend* command. There is no harm in regenerating messages, as ldb will ignore duplicate messages. If you and your opponent have forgotten whose move it is, the easiest way to get going again is for both of you to resend your last message. The one that got lost will be accepted, and the game will be off and running again. So that you can resend the last move of a game, completed games are kept in your .ldbdata for 1 week, after which they are deleted. They may be deleted sooner with the Delete Game command. Note also that games that have been waiting for your opponent for more than 1 week have an automatic resend performed whenever you run ldb.

Get Resend
: This allows you to send a message to your opponent's ldb requesting a resend. This saves your opponent the trouble of getting into *–control,* finding which game you're talking about, and resending. Instead, he can just feed your resend request message into ldb like it was a regular move. Version 1.0 of ldb not only does not support this feature, but will crash if a resend request packet is sent to it, so *–control* will check to make sure your opponent is running the proper version before sending the message.

Screen Dump
: This command makes an exact copy of the screen in the file "ldb_screen.dmp". This could be useful if you want to mail the board to someone or print it out.

Delete Game
: This command marks a game for deletion. The game is not deleted until you quit, and you can undelete it by pressing D again. Normally, you should not delete a game in this manner, you should play until someone wins or concedes. The *Delete Game* command can be used to get rid of games where your opponent doesn't answer his mail, or doesn't want to play, etc.

Next Game
: The next game is displayed. If there is no next game, this choice is equivalent to *Quit.*

Quit
: This allows you to exit from *–control* . If you made any changes to your games, you will be given the opportunity to save these changes, then ldb will exit.

VMS NOTES

VMS users have a few unique considerations:

On VMS, the default file names are changed from .ldb* to ldb.*. This groups all ldb files together.

VMS mail doesn't append messages to an existing file, it creates new versions. To accommodate this, vms-compiled executables will search all versions of a file by default. This applies to files specified on the command line as well as the file specified in your ldb.rc. You can disable this in either case by specifying a version number, or by specifying version number 0 to get the most current version. The default setting in ldb.rc is to scan sys$login:ldb.rcv;*. You might want to change this to ldb.txt;*, since that is the default extension added by mail when you extract a file. If you use multiple versions, you should be careful not to PURGE until after you have run ldb.

If you have the Eunice package, ldb will compile and run under it fine, but any files created by the Eunice–compiled version (most notably the games file) will be incompatible with the VAX C–compiled version. Use the unixtovms command to convert.

Ldb is available in vms_share format with the manual preformatted. If you have MMS, you can use the makefile.vms provided by entering:

```
$ rename makefile.vms makefile.
$ make
```

If you don't have a make available, you can compile with:

```
$ @makeldb
```

After ldb is compiled, you'll probably want to define a symbol such as:

```
$ ldb :== $dua0:[me.ldb]ldb
```

so you can run it.

The VMS implementation spawns a subprocess with no command when the superkey is pressed. You can enter LOGOUT or EOJ to return to the game. If you define a command to run when the superkey is pressed, the command will be run, but you will still need to logout of the subprocess to return to ldb.

BUGS

A game file created by ldb under Eunice must be run through "unixtovms" before the VAX–C ldb can read it.

The "Concede" command violates the rules, in that it allows one player to unilaterally settle the game for the current game value. This could be used to avoid an impending gammon.

The superkey function only works when ldb is waiting for a key. If ldb is doing something time-consuming, such as calculating legal moves, the superkey will not work until the calculation is complete.

There should be a way to play interactively (using socket connections, for example), rather than via e–mail.

Under Eunice, there is a bug when delmail is enabled, received mail is in a file with multiple versions, and these versions are entered on the command line (as opposed to being specified by a pattern in .ldbrc). Eunice handles multiple versions by appending a dot and the version number to all versions of the file but the newest version, to which it appends nothing. When the most recent version is deleted, the next older version becomes the most recent, and has no version suffix attached. This file will be skipped, since its name changed between the time the file list was created and the file is opened. All other files will be scanned properly. The work-around for this is to not use a pattern like 'ldb ldb.txt*' on the command line to specify all versions of the file, but rather use 'ldb ldb.txt.* ldb.txt' so that the most recent version is scanned (and thus deleted) last. Alternatively, the pattern can be quoted so that ldb interprets it instead of the shell.

The *–myaddr* command line option should override the myaddr line in the people file.

Version 1.0 of ldb only allowed the colors red, white, and black, represented by r, w, and b, respectively. Although ldb 1.0 will allow games to be started and played correctly with different colors, the move blocks will have the string "*BAD COLOR*" displayed where the color should be, making it difficult to figure out which color you are playing. Upgrading to a newer ldb will fix this. Nevertheless, if you are playing someone who is using ldb 1.0, you may want to restrict your color choices to r, w, and b.

The Crawford rule option may be incorrect. As implemented, it prevents doubling for any game where either player is within one point of winning the match. Some people play the Crawford rule such that doubling is only illegal for the first game that a player is withing one point of winning; if that player does not win the Crawford rule game, doubling for subsequent games is allowed. An authoritative description of this rule is needed.