Table of Contents

CHAPTER 1

Introduction

This is a user's guide for SRC Sd. If you are interested in basic start-up and running of Sd, read the first portion of each section, until you come to a phrase like, "you can skip the rest of this section until later." The basics of each section are presented first, and more subtle items later in the section.

Challenge callers may find this guide more useful than others, especially higher challenge callers, who often wish to use many stacked concepts. Getting the call desired at the lower levels is often easier than getting a lengthy call entered properly at C4. However, the basics of running Sd are level independent, and thus this guide is for everyone.

Sd will not necessarily help you become a better caller, or to call levels you don't already call. It will help you write material faster and more accurately, but it will not teach you about flow, timing, judging difficulty, sight calling, floor interaction, etc., all of which are important to your success as a caller. Also, we have seen callers attempt to use Sd to call levels they didn't know and end up with many inappropriate calls (either much too hard or just not worded the best way), which they couldn't explain because they didn't know how to do the calls either. We suggest that you only use Sd to automatically push checkers through calls that you could push checkers through yourself if you had to.

The authors are interested in making their product as accurate and useful as possible, and sincerely want your comments, compliments, and bug reports (including anything you want Sd to do that it can't seem to do). Please contact the authors with any of the above:

Sue Curtis
src@hnc.com
9454 Riverview Ave.
Lakeside, CA 92040

Lynette Bellini
lynette@ics.uci.edu
1300 Adams Avenue #6-L
Costa Mesa, CA 92626

1.2 Comments about the User's Guide

This user's guide is in an incomplete state, and will be for a long time. Any comments or suggestions for additions or changes will be most sincerely appreciated. Please direct such discussion to:

Lynette Bellini
lynette@ics.uci.edu
1300 Adams Avenue #6-L
Costa Mesa, CA 92626

1.3 The Completing Reader

SD features a completing reader, so named because it will complete any word you partially type in, so long as it is unambiguous. For example, at mainstream, sw th yields swing thru. Many of Sd's commands are verbose, e.g. allow modifications, but with the completing reader, it is only necessary to enter allo. Also, as soon as you type something the reader doesn't recognize, it beeps as soon as you hit the space bar.

1.4 How to Read this User's Guide

In order to make what is said be unambiguous, some conventions are followed. Items in bold are the items under discussion. Items inset and in bold are to be entered in Sd. Items in italic are call names. Items enclosed in <>, e.g. sdtty <level> are to be replaced with an appropriate item: sdtty c2. The angle brackets are not to be typed unless specified. This is not true in Sd itself, where angle brackets often must be literally typed.

CHAPTER 2

Starting Sd

Change to the directory where Sd resides. To just start the program at any level, type:

```
sdtty <level>
```

The level can be one of the following: m (mainstream), p (plus), a1, a2, c1, c2, c3a, c3, c3x, c4a, c4, or c4z. The default is mainstream.

The levels follow the Callerlab guidelines in general, although there are some calls that can be done from different formations at different levels (e.g. rotates can all be done at C1, although rotates from different formations are on different levels. Currently, there are no calls at the C3X level as this is no longer a used level. C4A calls are those on Lynette Bellini's C4 list, in general. C4 calls are those in use, but not by Lynette Bellini. C4Z contains calls that are no longer in use but have not been deleted from the database.

As soon as the program is started, Sd will show a line saying, "output file is: sequence.<level>" and then a line saying, "Enter new output file:". If you just hit return, the name of the file the sequences are stored in will be sequence.<level>. If you want a different file name, type the name you want. Remember there can't be any spaces in the name, and there is a maximum of eight letters, a period, and then three more letters. Next it will ask for a label to appear on all sequences - this will print on the second line of every sequence you write. I usually use the name and date of the dance. At the end of each sequence you will have the opportunity to change this label. If you do change it, what you change it to will be the default label until you change it again. If you don't want to change it, just hit enter. Next it will ask for a starting sequence number. Give it whatever number you want (if you give it nothing it will start numbering from zero). It will put the sequence number just after the label on each sequence. It will automatically increment the number for each additional sequence.

If you want to get right into writing material, you can skip the rest of this chapter. Don't forget about the rest of the chapter, though, because there are some useful tools here.

2.1 Command Line Arguments

There is a standard setup for Sd which is enacted when you start the program. To change some features, `command line arguments' are used - additional words, with a dash preceding them, after sdtty <level>.

2.1.1 Changing the Calls Database

To change the place where SD looks for the calls database (it normally looks in the file sd_calls.dat), the command line argument `-db <filename>' is used, as follows:

```
sdtty <level> -db <the name of your file>
```

This will only work if you have either changed the name of the calls database from sd_calls.dat to another name, or if you have created your own calls database under a different name.

2.1.3 Changing the Appearance of Sd

There are also some command line arguments that will affect how Sd appears on your screen. They are:

-no_cursor: tells Sd not to use cursor motion commands to update the screen efficiently (you generally won't want this unless for some reason Sd isn't displaying properly on your screen).

-lines <number>: changes the number of lines Sd displays at a time. The default is 25; if your screen is large enough for more lines, you may want to use this.

-no_line_delete: tells SD not to use the insert or delete line cursor control commands (you generally won't want this).

2.1.4 Abridgement and Printing out Call Lists

A nice feature for callers who are teaching classes is the abridgement feature. Sd recognizes two command line arguments that are used together to create a `class level' on SD. The first thing to do is to create the list of calls SD has on the level you are teaching. To do this, start Sd in the following manner:

sdtty <level> -write_list <file you want Sd to put the list in>

Sd will write the calls it has in its database at that level into the file you indicated. As you teach the class, delete the calls that you have taught from the list. Don't use a word processor, such as Word Perfect, since it inserts additional characters into files and makes them unrecognizable to Sd. Use a basic text editor, like DOS `edit'. To use the list you have generated, use the -abridge flag on the command line as follows:

sdtty <level> -abridge <file the list is in>

The <level> used must be the one the class is learning. Keep in mind that what you have created is a file that will allow Sd to start up at the given level, but exclude the calls on the abridgement list. Here is a real life example:

I'm teaching a C3 class on Wednesday nights. Before the class starts, I create the list of calls they will be learning:

sdtty c3 -write_list wednesday

For the first night, I delete from this file only the calls they will be learning the first night, and use Sd to write my sequences:

sdtty c3 -abridge wednesday

The next week, I delete the calls they will learn that week, and so on, until the file is empty and the dancers know all the calls.

You can't change the way calls look in the file used - they must match exactly the way Sd has them in its database. Also, all concepts at the level will be legal, and may be shown in resolves, so be careful to make sure they don't sneak in before you have taught them.

If you want to know exactly what calls Sd has at any given level, you can use the -write_list command to print out the list. This may be helpful if you know a call should be on a given list, but can't figure out the precise way the authors of Sd entered it into the program.

2.1.5 Writing out the Whole List

A command similar to -write_list is:

-write_full_list <file you want Sd to put the list in>

This puts every call Sd will use at the level given, including all lower level calls. I haven't yet found a use for this as it generates a huge list of calls, most of which are uninteresting.

CHAPTER 3

Sd Commands

The first command (not a call or concept but command to Sd) to learn is the !, which will list all the things you can do. If you've just started up Sd, type a ! now, and see what it does. You'll see a short list of things you can do:

exit from the program

```
     heads 1p2p
     sides 1p2p
     heads start
     sides start
     just as they are
```

The first choice will just dump you back out of Sd. You'll want to remember exit even though now isn't the time to use it, because that's how to get out when you're finished writing material. The next two choices (heads/sides 1P2P) start you in the heads/sides step right to a line formation. The next two (heads/sides start) move whichever dancers you chose into the center, ready to take the next call you give them. Just as they are keeps the dancers in a 16 matrix, where you can call all 4 couples or all 8 calls, or phantom lines/columns calls.

Choose either heads or sides start to begin with, then give the call you want them to do. Successive calls may be given just as calls, which will be dealt with later.

If you are anxious to start writing, go ahead and experiment by giving Sd some calls to do. When you are ready to resolve the sequence and end it, go to the section entitled The Resolve Command (section 3.6). There's more good stuff following, though, so please come back to it.

Following is a list of commands to SD that are not calls:

```
     exit the program
     undo last call
     abort this sequence
     insert a comment
     change output file
     end this sequence
     resolve
     reconcile
     create <formation>
     pick random call
     pick concept call
     pick simple call
     pick level call
     normalize
     standardize
     keep picture
     refresh display
     allow modifications
     simple modifications
     toggle concept levels
     toggle collisions
     ?
     !
```

3.1 Function Keys

Most of the above commands are available on the function keys. Following is a listing of what each function key does:

Key	Function Key	Shift	Control
F1	heads start	sides start	just as they are
F2	two calls in succession	twice	
F3	pick random call	pick concept call	pick simple call
F4	resolve	reconcile	normalize
F5	refresh	keep picture	insert a comment
F6	simple modifications	allow modifications	
F7	toggle concepts	toggle col/act phan	
F8	<anything>	<concept>	
F9	undo	abort the search	
F10	end this sequence	change output file	
F11	pick level call		
F12	find another	accept	previous

3.2 The !

The ! is doubtless the most important command, as it can let you know what is available to you in an unfamiliar area of the program. It can also help you find how a call is entered (Sd may

spell a call name slightly differently than you do), or just let you know what calls/concepts/commands start with those letters. We used the ! above to find out what was available upon entry to the program. Since there were a limited number of options available, ! was highly useful. If you try it again, though, the program will list every call and every concept and every command it knows. Unless you are working with a very small list (like the mainstream list), this will be an overwhelming output.

To view the output, keep hitting the space bar - this will step you through the output one page at a time. Hitting the return key will show one line at a time. When you're tired of looking through what Sd knows, hit the delete key. You will be left with the last part of the list you were looking at, plus the --> prompting you for another call. To get your screen back, try the refresh command (see section 3.4).

3.3 The ?

The ? acts as a more intelligent !. It actually tests each call and verifies that the call can be done from the formation the square is in, and with any concepts that have been entered. This means that the list of calls will be shorter, and Sd can do everything on the list. After the possible calls are all listed, Sd lists all the concepts. Remember, delete stops the listing.

Because the ? is intelligent, when Sd displays, "there are 3 matches; type ? for a list," it is possible that nothing will show. The three matches are syntactic matches - Sd can find three calls that start with the same letters you typed in - but the ? checks each to make sure those calls can actually be done from wherever you are. If you really want to see the syntactic matches, use the !.

3.4 The Refresh Command

Typing refresh causes the screen to go back to where you were before you asked for the list of calls. It will also work if your terminal puts out stray characters, or if you are using a modem that occasionally puts strange things on your screen. If for any reason Sd is having trouble keeping the screen in appropriate shape, you may want to use the -no_cursor flag described in the previous chapter. Doing so will cause a choppy appearance on the screen, and refresh will also fix this.

Refresh is F5 - press F5 and refresh will automatically be typed. Then hit return.

3.5 Warning

The following four commands: resolve, reconcile, pick and normalize, are simultaneously the most helpful part of the program and the most dangerous. Due to known misfeatures and unknown bugs, the program can and will produce illegal calls. Some of these misfeatures are very difficult to fix, and although the authors are aware of them, they haven't yet solved the problems. Others are actual bugs. Any time the program does something illegal, you should report it to the authors; their email addresses and physical addresses are listed in chapter 1.

The other problem with these commands is that Sd has no concept of difficulty. It doesn't know that combinations of concepts and calls can be downright impossible for real people to do, and happily presents them for you to select. Because they look neat, sound neat, or bring all the real people together again, you may be tempted to choose them. If you can't visualize what Sd did to end where it did, get out your checkers and figure it out. It is possible that Sd is wrong; or that the call is so hard you can't do it (in which case you had better not call it). Don't be tempted to say, oh yeah, I see it - make sure you can actually visualize what has happened - otherwise you could be very embarrassed.

3.6 The Resolve Command

The resolve command starts what you can think of as a separate program, whose job it is to find calls that resolve the square. This subprogram has its own set of instructions:

```
abort the search
find another
next
previous
accept current choice
raise reconcile point
lower reconcile point
quit
undo
```

as synonyms for abort the search. You can see these commands by typing resolve and then typing a ? or a !.

If you are just trying to resolve the sequence, type resolve, and find another until you see a resolve you like. Then type accept. Your sequence won't be finished yet - type end this sequence. You must be in an 8 person formation to use resolve, (i.e. no phantoms).

This is all you need to know if you're just trying to get out of your first attempt at using Sd. Please continue reading, though, as there're plenty of good tricks ahead.

Resolve is on F4. Just type F4 and resolve will be typed automatically. Then hit return.

3.6.1 How Resolve Works

The resolver works by choosing from one to three calls at random, and checking to see if they end in a right and left grand, left allemande, promenade, squared set, reverse promenade, single file promenade, or reverse single file promenade. It will try doing this up to 5000 times. If it doesn't find a series of calls that resolve the square in that number of attempts, it will return to you with the message "failed." This doesn't mean that there are no resolves, it only means that after trying 5000 random combinations, no resolves were found. You can try find again after it has come back "failed," and it may well find a resolve the next time.

You can first enter a concept, and then use the resolver. This will cause the first call produced by the resolver to be modified by that concept. It doesn't guarantee that there will be a one call resolve using that concept.

The resolver works in asymmetric formations, but realize that choosing a call or resolving the square in an asymmetric sequence is very difficult for SD since it generates calls randomly. So, don't expect much if you try resolve from an asymmetric formation.

Reconcile, pick and normalize work in the same, random way.

3.6.2 Abort the Search (Quit, Undo)

Abort, quit and undo take you out of
resolve mode and puts you back to a call prompt. This is useful if the resolver isn't finding anything you like. Rearrange the dancers by calling another call and try again.

Abort is on Shift F9. Hold down the Shift key and press F9. Abort the search will be typed automatically. Then hit return.

3.6.3 Find Another

Find another asks the resolver to produce another resolve for you. It will do this as many times as you ask, until it can't find any more resolves (and then it will just keep returning `failed'). Remember you can just type f and the completing reader will type the rest.

Find Another is on F12. Press F9, and Find Another will be typed automatically. Then hit return.

3.6.4 Next and Previous

After you have generated several resolves, you may decide that you are tired of searching for a resolve, and that one of the first resolves Sd produced is good enough. Typing previous steps back through the resolves. Do this as many times as you need to get back to the resolve you want. Once you get there, you may decide that it is not as nice as one of the later resolves. Move the other direction by typing next until you get to the resolve you want. You can type find at any point, and it will generate a new resolve. It will also take you back to the end of the resolves, and all previous resolves will be found by typing previous. You can think of the resolves as being listed on a sheet of paper. Each new resolve is put at the end of the list. The previous command moves your eyes to the previous line (if there is one), next moves your eyes to the next line (if there is one), and find always adds one to the bottom of the list and moves your eyes there.

If you've generated a hundred or so resolves, using this method may be time consuming. If you can remember the getout you liked, you can always abort the search and then just enter the calls yourself.

Previous is on Control F12. Hold down the Control key and press F12. Previous will be typed automatically. Then hit return.

3.6.5 Accept

Accept means that you want the resolve you are currently looking at; it enters those calls into the list of calls you have already entered in your sequence, and returns you to the call prompt, showing "resolve is:" and the out (right and left grand etc.) you resolved to. To end the sequence you must type end this sequence.

Accept is on Shift F12. Hold down the Shift key and press F12. Accept will be typed automatically. Then hit return.

3.7 Reconcile

Reconcile is a variation on resolve that is way of finding a way to use a clever getout that didn't quite work. If you've done some neat call, and want to end the sequence on that call, but the dancers are not set up appropriately, use reconcile. Reconcile can only be used from left-handed two-faced lines, right-handed waves, or left-handed waves, for which it assumes you want a promenade, right and left grand, or left allemande respectively. This is so it can tell what kind of ending you want. If you have an 8 chain thru formation, it can't tell whether you are looking for a left allemande or a right and left grand ending. You can do a touch or a left touch to give SD a hint, then remove it later.

Reconcile uses the same commands as resolve, but there are two commands specifically for reconcile, raise and lower. When you first ask for reconcile, your sequence will appear, with a dotted line through it. This is called the insertion point, and it is where SD will attempt to put calls to make your neat getout work. To move it up one call, use raise, and to move it down one call, use lower. When you have the insertion point where you want it, type find. If reconcile can't find anything after several attempts, try changing the insertion point. It won't work well from obscure setups for the level you are writing (hourglasses at mainstream are an example).

It is better to avoid having a gender dependent or head/side dependent call between the insertion point and the end of the sequence. It will be much more difficult for SD to find an appropriate reconcile with this situation.

Reconcile is on Shift F4. Hold down the Shift key and press F4. Reconcile will be typed automatically. Then hit return.

3.8 Pick Random Call/Concept Call/Simple Call/Level Call

Pick Random Call selects a random call. It works with the same commands as does resolve and reconcile. The call may or may not have a concept with it. You can enter a concept first, then pick random, and it will choose calls with that concept. If there are no calls that can be done with that concept from the formation you are in, it will fail repeatedly. By entering the concept you want, followed by pick, Sd will generate every call that can be done with that concept if you type find another enough times. If that concept you choose is not legal from where you are, no warning will be produced - Sd will just keep returning with the message, "failed."

Pick random is on F3. Hit F3, and pick random will be typed automatically. Then hit return.

With pick concept call, Sd chooses a concept and a call. With pick simple call Sd only selects calls, without the chance of a concept being selected. With pick level call SD only selects calls at the level entered at the time you started the program (if you started with "sdtty c2," then pick level call will randomly present calls it has at C2).

Pick concept call is on Shift F3. Hold down the shift key and press F3 and pick concept call will be automatically typed. Pick simple call is on Control F3. Hold down the control key and press F3 and pick simple call will be automatically typed. Pick level call is on F11. Type F11 and pick level call will be automatically typed.

3.9 Normalize

Normalize is useful from non-ordinary formations (those containing phantoms), and also with concepts. From non-ordinary formations, it searches for calls/call-concept combinations that return all real dancers to an ordinary formation. With a concept, from either an ordinary or a non-ordinary formation, Sd uses the concept for at least the first call, and ensures that all live dancers return to an ordinary formation. A good example of using it with a concept is to Own Someone for Something by normalize. You choose the someone and the something, then normalize will find a call for the other dancers that will leave you in an ordinary formation.

Normalize is on Control F4. Hold down the control key and hit F4, and normalize will be typed automatically. Then hit return.

3.10 Standardize

Standardize works somewhat like normalize. It takes a t-boned non-phantom formation and presents calls that yield a non-t-boned, non-phantom formation.

3.11 Create <formation>

From any 8 dancer formation, typing create <formation>, where the formation is one of the following, will cause Sd to present calls that end in the specified formation. It works just like any of the "picks." The following formations are currently available:

- any tidal
- any lines
- any columns
- any 1/4 tag
- lines in
- lines out
- 3x1 lines
- inverted lines
- 2fl
- waves
- columns
- magic columns
- 8 chain
- trade by

dpt
cdpt
1/4 tag
3/4 tag
1/4 line
3/4 line
diamonds

If you think there ought to be other formations available, please let us know.

3.12 Change Output File

This command changes what file the sequence will be written to. If you didn't use the -sequence option (explained in 2.1.2), or change it at program start-up, the default is sequence.<level>. Once you're in the program, this command allows you to change the file name. You can enter a full path name here, so if Sd resides in C:¥SD, but you keep your sequences in C:¥sequences, you can say change output file, then give it C:¥sequences¥mysequence for a file name. Sd doesn't append any extents (the "txt" of sd_doc.txt is an extent) with this command (it does if you use -sequence at start-up), so you can use precisely the name you want for your file.

Change output file is on shift F10. Hold down the Shift key and hit F10, and change output file will be typed automatically. Then hit return.

3.13 Keep Picture

Keep picture will save the last picture into the file for you. This is especially useful if you just called something difficult or that leaves people in a very large matrix. If you are not good at watching the dancers and knowing where they should end, or just visualizing what should be happening, this feature may help you.

Keep picture is on Shift F5. Hold down the Shift key and hit F5, and keep picture will be typed automatically. Then hit return.

3.14 Undo

Undo undoes the last call entirely, concepts and all. If you have a call partially entered, with a couple of concepts, for example, undo will remove the last concept.

Undo is on F9. Hit the F9 key and undo will be typed automatically. Then hit return.

3.15 Insert a Comment

Insert a comment allows you to write a note to yourself before any call. Be aware that this command is attached to whatever call follows it, however unnatural this may seem. If you want your comment to be the only thing on the line, follow it with the call nothing. You can use this for any purpose, cues,
or, if you have used some miscellaneous calls to get Sd to do a call it couldn't do alone, to enter the name of that call. If you do this, be very careful when editing the sequence. Almost all errors attributed to Sd are actually editing errors.

Insert a comment is on Control F5. Hold down the control key and press F5, and insert a comment will be automatically typed. Then hit return.

3.16 End this Sequence

This command can only be used when Sd is displaying the message, "resolve is: xxx" at the bottom of the sequence. If that isn't there, the square is not resolved, and Sd can't end the sequence. It doesn't matter how the square was resolved; whether with the resolver, by accident, or any other method, so long as the square is resolved, that message will be displayed. Once it is displayed, use end this sequence to end the sequence.

Sd will ask if you want to enter a different label; just hitting return will leave the label you chose for all sequences at program start-up. This is the point at which the sequence is actually saved to the disk. Then, it will display the sequence to you along with a new start-up prompt, ready to begin the next sequence.

End this sequence is on the F10 key. Hit F10 and end this sequence will be typed automatically. Then hit return.

3.17 Toggle Concept Levels

This is to enable usage of concepts at levels below their actual level. It is common usage to have boys do one thing and girls do another at all levels. A convenient concept that does this is own the <anyone>, but it is a C3A concept. If you want to use it at C1, use toggle concept levels to have access to own the <anyone>, enter the call, and type toggle concept again. This returns the concept levels to their proper state. If you leave Sd in the `all concepts' mode, any resolves will use all the concepts. You can tell if it is in the "all concepts" mode by looking for a [ac] before the > in the call prompt. To remove it, type toggle concept again (it works like a toggle switch - the light is on or off).

Toggle concept levels is on F7. Hit F7 and toggle concept levels will be typed automatically. Then hit return.

3.18 Toggle Collisions

The purpose is to make Sd better at allowing people to come to the same spot. If you want to do a call that ends with people on the same spot, and Sd won't let you do it, try toggle collisions. If [col] is showing before the > in the call prompt, collisions is turned on. To remove it, type toggle collisions again (it works like a toggle switch - the light is on or off). It is quite possible that Sd still won't be able to do what you want. Its ability to tolerate collisions is pretty low.

Toggle collisions is on Shift F7. Hold down the Shift key and hit F7, and toggle collisions will be typed automatically. Then hit return.

3.20 Simple Modifications and Allow Modifications

If a call can be modified in some manner, e.g., the star can be turned an extra fraction, or you want to replace some part of the call with something else, one of these two commands will allow you to do it. Simple modifications have been somewhat arbitrarily defined as those for which there is an accepted way of fitting the words in without using the phrase "but replace the <whatever> with <whatever>."

Before entering the call, type allow/simple modifications (whichever is appropriate). If the call can indeed be modified, Sd will tell you what part of the call can be replaced, and ask you if you want to replace it. If you do, type a y, otherwise type an n. If you typed y, it will then ask you for the replacement call. This replacement call can have any concepts on it that are appropriate. Some calls may have many places where it can be modified. You can select any or all of these modifications. Sd will display these modified calls using brackets in places that help make their meaning clear.

Simple modifications is on F6. Hit F6 and simple modifications will be typed automatically. Then hit return. Allow modifications is on Shift F6. Hold down the Shift key and hit F6, and allow modifications will be typed automatically. Then hit return.

3.21 Exit

If you are at the start-up prompt, exit takes you out of Sd immediately. If you are partially through writing a sequence, it will warn you that the current sequence will be aborted, and ask you to verify that you want to exit by typing y. Then Sd will

exit.

CHAPTER 4

Entering Concepts

In this chapter, examples are presented inset and bold. Items on separate lines indicate that there must be a return, or enter, between typing each line.

Concepts are entered by themselves, preceding the call. For example, if you wanted the call tandem walk and dodge, it would be entered as follows:

```
tandem
walk and dodge
```

and they may be stacked, one after the other, so long as there is a return (or enter) after each concept.

This is all you really need to know if you're just trying out Sd for the first time. This chapter is for those who are trying to get Sd to do concepts that it won't seem to do. The chapter is organized by alphabetical listing of concepts. If a concept is not in this section, it means that it is entered into Sd in a straightforward manner in the way that you would call it, and that it doesn't have any special problems (like it can't do certain forms of the concept).

It is crucial to enter concepts one per line, so to speak. Any time Sd beeps at you when you are trying to enter concepts, it is likely that you've put a couple of concepts together. It may not always be obvious what a concept is (see section 4.11 on Phantom Couples/Tandem (Twosome)). We have a report of a user, trying to get split phantom lines, who typed:

```
split
phantom
lines
```

This is extreme. The concept is split phantom lines. However, keep in mind that it may not always be obvious what is being considered a concept - try it different ways.

4.1 Assume Formations

These concepts exist in order to tell Sd which way phantoms are facing in phantom formations (see section 4.12 on phantom facing directions, which gives a much better explanation of the entire problem). They are used as follows:

```
split phantom columns
assume normal columns
turn the key
```

Without the assume normal columns, Sd doesn't know which way the phantoms are facing, and consequently doesn't know how to do the hinge at the end. Any time you get messages that complain that Sd needs real people, not phantoms, or that it doesn't know how to cast, or that it is losing phantoms, use assume to tell it which way the phantoms are facing. Following is a listing of formations that can be assumed.

```
assume symmetry
assume couples
assume miniwaves
assume waves
assume 2fl
assume 1fl
assume inverted lines
assume normal columns
assume magic columns
assume 8c/trade by
assume dpt/cdpt
assume 1/4 tag
assume 3/4 tag
```

```
     assume 1/4 line
     assume 3/4 line
     assume normal diamonds
     assume interlocked diamonds
     assume facing diamonds
     assume facing intlk diamonds
     assume left 3/4 tags
     assume right 3/4 tags
     assume right 1/4 lines
     assume left 1/4 lines
     assume left 3/4 lines
     assume right 3/4 lines
     assume normal diamonds
     assume facing diamonds
     assume normal interlocked diamonds
     assume facing interlocked diamonds
     assume normal casts
```

4.2 Centers

Since dancers can have problems understanding which centers the caller is discussing, it is no wonder that it is a difficult concept to program. There are certain formations from which the word centers is ambiguous, and this section will attempt to tell you how to activate the dancers you want. In some cases, it's just not currently possible.

From a tidal line, there are two possible sets of centers: the centers of each 1x4 (each line), and the center 4 of the formation. Just typing centers activates the center 4 of the formation. To get the centers of each 1x4, type:

```
     each 1x4 (each wave and each line are synonyms for each 1x4)
     centers
     any call (just the centers of each side will do it, so
     make sure it is a 2 (or 1) person call.
```

From the crossed wave formation:

```
     3G>
     2G<
4Bˆ 3BV 1Bˆ 2BV
     4G>
     1G<
```

the only centers available are the center wave (the boys in the above diagram). There is currently no way to call center diamond from here.

If you want the centers of a parallelogram to work:

```
     parallelogram
     centers
     walk and dodge
```

and in the parallelogram, centers will do the walk and dodge.

The center phantom setups can be identified, e.g.:

```
     center triple box
     walk and dodge
```

or

```
     center phantom waves
     acey deucey
```

4.3 Central

In general, SD knows central as a concept, so it is entered as:

```
     central
     detour
```

However, there are some calls that can be done central that SD knows just as calls. So, if SD can't do what you want, try it all on one line:

```
     central spin the windmill
```

4.4 Cross

There is nothing unusual about the cross concept, except that it doesn't seem natural to many people to enter cross separately from the call it is modifying. Cross is a concept, and it must indeed be entered separately from the call it modifies, e.g.

```
     cross
     invert the column
```

There are some calls where cross is part of the name of the call (e.g. cross and wheel, crossfire) and these are entered just as calls are normally entered.

4.5 Disconnected

SRC Sd doesn't do disconnected at all (yet; we're hoping for the winter release).

4.6 Ends

There is an unfortunate but well known bug associated with the ends. Sd views the ends as a virtual real group of four (a box or line, usually). Consequently, Sd will permit and suggest (with pick r/s/c) calls for just the ends that require the ends to work in a box of four. Try:

```
     heads start
     pass in
     ends
     pick simple
```

You should generate some interesting things; I immediately got ends wheel fan thru, and ends vertical left 1/2 tag. This problem can occur from either lines or columns. Beware of pick.

The other result of Sd's view of the ends formation is that asking the ends to do an 8 person call usually doesn't work. You must say, for example:

```
     ends do your part (ends dyp is a synonym)
     grand chain 8
```

Naturally, the problems that occur with identifying centers also occur with identifying the ends. From a tidal line, there are two possible sets of ends: the ends of each 1x4 (each line), and the end 4 of the formation. Just typing ends activates the end 4 of the formation. To get the ends of each 1x4, type:

```
     each 1x4 (each wave and each line are synonyms for each 1x4)
     ends
     any call (just the ends of each side will do it, so make sure it is an          appropriate call.)
```

The outside phantom setups can also be identified, e.g.:

```
     outside triple boxes
     walk and dodge
```

or

```
     outside phantom waves
     swing thru
```

4.7 Finish

This paragraph is not about Sd. Rather, it is to clarify the finish concept. Finish was an ill-defined concept for some time, which meant to do however much of the given call the caller wanted you to do. This was difficult for dancers to do, and a "real" definition has replaced it at the higher levels. This definition may not be known to callers who don't do the higher levels, but this definition is how Sd does finish - which is why this paragraph is here. The precise definition of finish is to do

all but the first part of the given call. Corruptions such as finish like a (e.g. wheel the ocean) are not tolerated by Sd, because they are corruptions (see section 4.10 on like a), meaning to do all but the first part of the last part!

4.8 Funny

Funny is a very difficult concept to program (as you will discover if you try and invent one consistent and useful definition of it), and consequently it is highly restricted in Sd. To get those "funnies" that Sd can do, just enter funny as you would any other concept, and then the call.

Funny square thru is a separate call and is entered just by typing the call as you would say it:

```
funny square thru 2
```

4.9 Interlocked Diamonds

Interlocked is a concept that applies to anything with the name interlocked, including interlocked diamonds. To get interlocked diamond calls, enter interlocked (followed by return or enter) and then whatever call you want, like this:

```
interlocked
diamond chain thru
```

This will be written into your sequence as interlocked diamond, diamond chain thru. There is no separate concept "interlocked diamonds."

4.10 Like a(n)

This paragraph is for clarification only. Similar to the finish concept, this concept started out as "do as much of the call as you think the caller wants you to do." This was difficult for dancers to do, and has been replaced by a "real" definition as follows: do the last part of the given call. Sd can't tolerate such corruptions as finish like a (e.g. wheel the ocean), because it is a corruption. It technically means to do all but the first part of the last part. SD will only do the real definition.

4.11 Phantom As Couples (Twosome) or Phantom Tandem (Twosome)

It seems unnatural, but phantom is one concept and as couples (twosome) or tandem (twosome) is another, and so they must be entered as follows:

```
phantom
as couples
any call
```

or

```
phantom
tandem twosome
any call
```

Just remember that every individual concept must be entered on its own line (so to speak). Phantom is a concept. One thing to note is that, if the live people are paired up with live people, Sd doesn't care if you use the word phantom or not - just tandem will have the same effect. The dancers will care, so use the word anyway.

If you are in a normal formation to start any of these calls, Sd assumes that you will use a 4x4 matrix for the next call. Consequently, if you are in columns, and want to call a phantom tandem twosome wind the bobbin, you have to tell it that it needs a 2x8 for that call. Enter it as:

```
2x8 matrix
phantom
tandem twosome
wind the bobbin
```

4.12 Phantom Couples (Twosome) or Phantom Tandem (Twosome) in a 1/4 Tag

To get any variation
of the above, enter the pairing (phantom as couples (twosome) or
phantom tandem (twosome)) first, as a concept, then the phantom
1/4 tag concept, then the call:

```
couples twosome
phantom 1/4 tag
straight away
```

Notice that it is phantom 1/4 tag, singular. From a tidal wave,
split phantom 1/4 tags is plural.

4.13 Phantom Facing Directions

Sd doesn't have any understanding of the facing direction of
phantoms built in to the ordinary phantom concepts. It doesn't
assume symmetry, or anything similar, that real dancers do all
the time. Consequently, if you try and call something like, from
normal right hand columns, split phantom columns turn the key
(hinge, counter rotate, trade), Sd will give you an error message.

It is of utmost
importance to communicate to the dancers what is expected of
them. Assuming that the phantoms are, when possible, symmetric to
the live dancers is generally accepted by the challenge
community. Forcing dancers to track, throughout a call, phantoms
positions that are dependent on the phantoms initial facing
direction is not generally accepted.

To get Sd to do split phantom columns turn the key, type:

```
split phantom columns
assume normal columns
turn the key
```

You can use assume symmetry instead of assume
normal columns. Remember that this only works so long as each
phantom has a live opposite in the phantom formation. You must
give a more specific formation (like assume normal columns) if
some phantoms don't have live opposites. See section 4.1 for more
on assume formations.

4.14 Reverse Random

Now that you are accustomed to entering each concept on a
separate line, here is a place where they are entered on the same
line:

```
reverse random
tandem
couple up (parts)
```

instead of the expected

```
reverse
random
tandem
couple up (parts)
```

4.15 Split Concept

There is no such thing as the "split concept." Any call that contains
the word split is entered just as you would say it:

```
split square chain thru
```

4.16 Split Phantom 1/4 Tags/Diamonds

Sd views these two formations as identical.
Sd can pick some strange
things with these concepts. From a tidal wave, for example, split
phantom 1/4 tags, pick r/s/c, will generate many calls that can

only be done from diamonds. Just beware that it is not capable of distinguishing between the two (and prefers diamond calls).

4.17 3x1 Triangles

At C4, 3x1 triangles may be entered as:

```
3x1
triangle circulate
```

At C2 and above, the call 3x1 triangle circulate is available as a call, typed exactly that way.

4.18 Those Facing (Trailers Start)

A commonly used call is those facing start a rotary spin. Sd doesn't know those facing, but it does know someone start. Those facing are generally trailers, so you can get the same result by:

```
trailers start
rotary spin
```

Of course, this works with any appropriate call, not just rotary spin.

4.19 Two Calls in Succession

This concept was originally intended to handle things like jay revolve to a wave like a recoil. Since Sd needs a concept in front of every call, and revolve to a wave and like a recoil are two separate calls, they must be entered on separate lines. But after the revolve to a wave, the jay is destroyed, and the ending formation is unknown. To get this call, do the following:

```
jay
two calls in succession
revolve to a wave
like a
recoil
```

SD isolates the people in the jay for both calls instead of one at a time.

This concept has also proven useful by providing a way to place two calls on one line. Since things like roll, spread, and with the flow are calls that naturally go with other calls, it is awkward to have the call and then one of these suffix calls on separate lines. Some examples of how to use this:

```
two calls in succession
recycle
with the flow

two calls in succession
follow your neighbor
spread

two calls in succession
swing thru
boys roll
```

Remember that you don't have to type out two calls in succession - just hit F2 and it will type it for you.

CHAPTER 5

Entering Calls

In this chapter, examples are presented inset and bold. Items on separate lines indicate that there must be a return, or enter, between typing each line.

Calls are entered by just typing the call name and return or enter. If Sd won't do the call and there could be a space somewhere in the call, try it with that space. If you have spaces and there shouldn't be one, the completing reader (see

section 1.3) will usually let you know. This is all you really need to know if you're just trying out Sd for the first time. This chapter is for those who are trying to get Sd to do calls that it won't seem to do. The chapter is organized by alphabetical listing of calls. If a call is not in this section, it means that it is entered into Sd in a straightforward manner in the way that you would call it, and that it doesn't have any special problems.

5.1 and Cross

and Cross is a call by itself, so anything like touch 1/4 and cross must be entered as two calls:

```
touch 1/4
and cross
```

Of course, you can use two calls in succession to get it to show all on one line:

```
two calls in succession
touch 1/4
and cross
```

5.2 Bridge the Gap

Sd can't do bridge the gap from any kind of phantom concept, like split phantom columns, bridge the gap. The call bridge the gap (dpt) is provided to allow this.

5.3 Cast Back (Cross)

People may be designated to cast back some direction, e.g.:

```
boys cast back (right)
```

It doesn't know ends cast back (e.g. from a trade by), but it can do leads cast back from a completed double pass thru.

5.5 Clean Sweep

If you want to fractionalize this call in any way, you must use clean sweep (parts). Also, finish clean sweep is a call; use it if that's what you want.

5.7 Dixie Style Calls

If you are trying to call dixie style calls from a beginning double pass thru (1x4), you have to use the concept single file to get the dixie style calls to work:

```
single file
dixie style to a wave
```

5.8 Easy Does It

In order to use some concepts with this call, a separate call was created, easy does it (lines). If you are having trouble getting it to do, for example, 3x3 easy does it, use the (lines) version instead.

5.10 Invert n/4

Many callers call the following set of calls frequently: from columns, invert 3/4, outside columns of 3 invert 2/3. This last call must be entered as follows:

```
invert 3/4
outer 6
invert (the column) 2/3
```

5.11 Kickoff

This is tricky to enter because you have to realize that you designate who kicks off and the call all on the same line. If you

are accustomed to typing centers (enter) to get the centers to do things, it won't work with kickoff, which must be entered as:

```
centers kickoff
```

Centers cross kick off must be entered all on one line, also:

```
centers cross kickoff
```

From a tidal line, there are special problems for Sd in understanding which centers are designated. To have each line do a centers kickoff, type:

```
each 1x4 (each wave and each line are synonyms for each 1x4)
centers kickoff (or cross kickoff)
```

5.12 <anyone> Pinwheel

This can only be called from setups where ends are defined. Specific people can't be designated; ends pinwheel must be used:

```
ends pinwheel
```

5.13 Plenty, replace the box circulates with...

To modify the star turns on plenty, you must use allow modifications.

```
allow modifications
plenty
y  (it asks whether you want to replace the star turn)
flip back  (it asks for the replacement call)
```

5.14 <anyone> Press

From a 1/4 tag formation, SD doesn't have a precise matrix (so it can do calls that require it to be in a 1/4 tag and in diamond spots), and in order to do any call that involves moving to exact matrix spots, like shove off, press, and truck you have to tell Sd what matrix to use on any of these calls:

```
3x4 matrix
head girls press right
```

5.15 Reverse Cut/Flip the Diamond

This must be entered as concept call:

```
reverse
cut/flip the diamond
```

5.16 Reverse Explode

Since reverse explode can be done from waves, which are legal at C1, and from lines, which aren't legal until C4, and Sd can't distinguish between the two, it will "pick" reverse explode from lines at C1. Beware of pick.

5.17 Rotates

Since rotates are spread across three lists, and Sd can't distinguish between the different types of rotates, Sd will present rotates from all positions starting at C1. Make sure you don't call them at an inappropriate level.

Also, there is a call, ends single rotate, to be used when you want the ends of columns only to do the rotate, as follows:

```
ends
single rotate 1/4
```

or

```
ends
reverse
rotate 1/4
```

5.18 <anyone> Shove Off

From a 1/4 tag formation, Sd doesn't have a precise matrix (so it can do calls that require it to be in a 1/4 tag and in diamond spots), and in order to do any call that involves moving to exact matrix spots, like shove off, press, and truck you have to tell Sd what matrix to use on any of these calls:

```
     3x4 matrix
     boys shove off
```

5.20 Spread and "and Spread"

Spread is a call by itself, and is the kind of spread that equals a slide. It is appropriate after calls like follow your neighbor, or peel the top.

and spread is a different call entirely, to Sd. It is the kind of spread that equals a slimdown. It is appropriate after calls like wheel and deal or heads pass in. This was recently changed from ... and spread.

5.22 Swing Thru (pu diamonds)

From a two-faced line, if centers hinge and roll, and you want them to start a swing thru (or left swing thru, as appropriate), you must use swing thru (pu diamonds). If it is left, just use the left modifier before the call, as always.

5.23 <anyone> Swivel/Cross Swivel

This call goes all on one line. boys cross swivel is entered exactly like that, all on one line.

5.25 Triangle Circulate

Simply naming the triangle you want, and then telling it to circulate will produce error messages. You must use the call triangle circulate instead of just circulate

5.26 <anyone> Truck

From a 1/4 tag formation, Sd doesn't have a precise matrix (so it can do calls that require it to be in a 1/4 tag and in diamond spots), and in order to do any call that involves moving to exact matrix spots, like shove off, press, and truck you have to tell Sd what matrix to use on any of these calls:

```
     3x4 matrix
     side girls truck
```