Trigonometric Proofs
An AROMA Case Study

Jason A. Collins

Supervisor: J. E. Greer

Abstract

Granularity hierarchies implement a method that has been used recently within the ARIES Laboratory for the recognition of entities under less than ideal conditions (e.g. incomplete information). Follow–up research including the addition of constraints, controls, clusters, contexts, and case–based reasoning has given the granularity hierarchy approach the power to be used in real–world recognition problems.

Two cooperative systems have been developed for the implementation of a granularity hierarchy in a real world context. Built for the recognition of solutions to recursive LISP or SCHEME programs, the SCENT system is an example of a working granularity hierarchy. SCENT is developed within the AROMA shell — a tool which provides case–based reasoning about student solutions. The PETAL system assists novice users in the development of these recursive LISP or SCHEME programs through the use of scaffolding; a powerful front–end that captures the typical mental models of novice programmers is used to help the students construct solutions without concerning themselves with the complexity of syntax. PETAL and SCENT (along with the case–based reasoning extension made available by AROMA) provide a powerful learning environment for a novice student.

Studies must be performed on the above systems to quantitatively determine their actual effectiveness. Unfortunately, the nature of the SCENT domain severely restricts the number of prospective test subjects in the pool. That is, there is a very small window of opportunity in which there are available test subjects with just the required amount of training to participate in a study about novice recursive LISP (or SCHEME) programmers.

In order to more effectively and accurately study these systems' effectiveness, a domain with a larger pool of subjects must be implemented. Trigonometric identity proofs represents such a domain. Many students have taken a course in trigonometry in secondary school but not beyond. Since high school gives an adequate level of training for this domain, it is a natural solution to finding a larger set of potential candidates for empirical study. This paper discusses the implementation details required for this domain and also performs a case study on the design of a new granularity hierarchy in general.

1 Introduction

Novice learning environments are a main focus of the ARIES Laborartory. Development of the PETAL and SCENT systems have been instrumental in the creation of these novice learning environments for novice recursion programmers in the domain of LISP and SCHEME. Empirical studies have been performed to determine the effectiveness of this learning environment, however these studies are weak

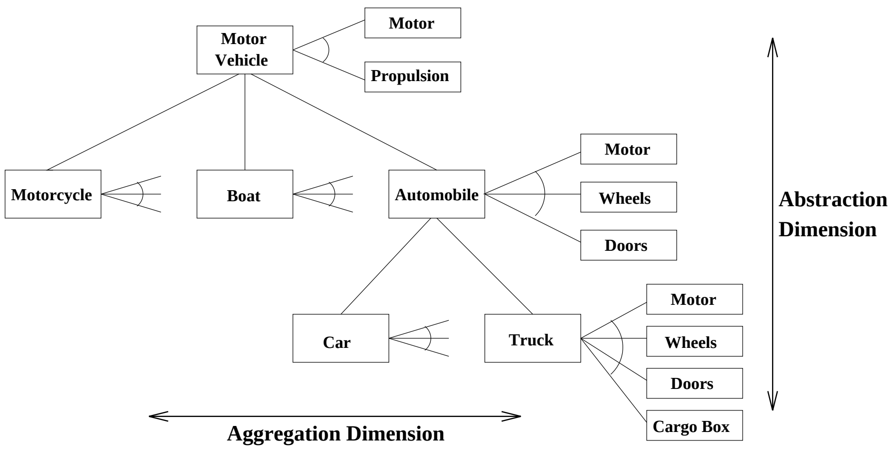

Figure 1: A granularity hierarchy for the domain *motor vehicle*.

solely due to the lack of test subjects. Test subjects are scarce becuase a very specific amount of training is required for the PETAL–SCENT domain and there is a very short period of time where a very small number of students have this exact amount of training.

A new domain must be found in which test subjects are plentiful. This paper describes the attempt to find that domain in trigonometric identity proofs. There is a large pool of test subjects with the exact amount of prerequisite training in this domain. If it is possible to implement this domain as a granularity–based learning environment, larger empirical studies can be performed to determine the actual effectiveness of the granularity approach to tutoring.

This paper formalizes the trigonometric identity proof domain, describes how it is knowledge engineered and implemented as a granularity hierarchy, and discusses the pitfalls encountered in developing a granularity hierarchy for a new domain.

2 Background

2.1 Granularity Hierarchies

Granularity refers to the level-of-detail or perspective from which a concept or entity is viewed.[7, 5] A granularity hierarchy captures different levels of detail in a type of semantic network. This semantic network identifies both abstraction and aggregation relationships in two orthogonal dimensions.[6]

Given a set of incomplete information, an entity can be recognized by the granularity hierarchy to as detailed a level as is possible.[8] Consider the granularity hierarchy in Figure 1 and the set $S = \{doors, wheels, motor\}$. Since all the objects in the aggregation dimension of automobile are present, we can say that we have recognized an automobile. This recognition is further propagated up the abstraction dimension allowing us to say that we have recognized a motor vehicle as well. Notice, however, that we can not say we have fully recognized a car or truck since we do not have enough information to allow that specialization.

In the presence of incomplete information, an entity can be viewed in the most specific manner possible through the use of granularity hierarchies.

2.2 The SCENT Project

The SCENT project uses granularity hierarchy techniques in the recognition of simple recursive LISP or SCHEME functions. The finest grained aggregation levels of the hierarchy observe portions of code specific to LISP or SCHEME syntax (e.g. `car` or `cdr`), while the internal portion of the hierarchy combines these primitives into recursive solution strategies. Using SCENT, student solutions to simple list–recursive problems may be recognized in terms of the strategies and concepts therein. This recognition identifies student solution strategies and aids in the delivery of advice to the student about possible errors.

The SCENT granularity hierarchy is implemented within AROMA and is composed of over 200 strategy objects. This hierarchy is used to recognize and "finger–print" (i.e. uniquely identify) any simple list–recursive LISP or SCHEME function.

For a given task (for example, the **`findb`** problem), a set of cases may be developed to form the case library. These cases represent typical optimal, sub-optimal, or incorrect solutions to the problem. Each case is recognized by the granularity hierarchy system to form an instantiated hierachy. The instantiated hierarchy may be tagged with advice to help students in areas of difficulty. These instantiated hierarchies are then stored in the case library to be used for later diagnosis.

When a student enters a solution, it is recognized (or finger–printed) by the granularity hierarchy. After this step is done, it is compared to the instantiated hierarchy of each case in order to find the one most similar. Once this match is found, the student may browse the provided advice to discover the differences between his or her solution and the system solution. This method forms a powerful advising system capable of giving advice and suggestions to a novice learner having difficulty.

2.3 The PETAL System

In many domains, student solution entry is a difficult problem. Frequently the student is concentrating so intently on the correct syntax and formulation of a solution, that it is no longer possible to concentrate on the solution itself![1] It is desirable to have a front–end to the solution entry that assists students not only in entering the correct syntax for a solution, but also in supporting the mental models required to correctly and efficiently solve a problem.

Bhuiyan's PETAL system accomplishes this task.[2] Continuing in the recursive LISP or SCHEME domain, PETAL allows students to formulate LISP or SCHEME solutions with correct syntax. Further, PETAL purports to assist students in the development of mental models aiding in problem solution. Various studies into the effectiveness of this system have been performed; see [2] for a summary.

Using a front–end such as PETAL can provide numerous advantages. First, the syntactical correctness of any solution can be guaranteed. Since a PETAL front–end scaffolds the correct syntax, incorrect formulations are impossible. Second, the PETAL front–end can filter out unneeded information. If, for example, knowing that there was a *cdr* reduction (e.g. `(cdr L)`) was unnecessary for recognizing recursive LISP or SCHEME functions, PETAL could simply not pass on that piece of information to the granularity recognition engine. This two step approach to interpreting solutions allows simplification of the granularity hierarchy, which is desirable both for the understandability of the hierarchy proper and for the computational efficiency of the recognition process.

2.4 Motivation

Studies have been performed to determine the effectiveness of the combination of the SCENT and PETAL systems.[9, 3] These studies have been very weak because of the small number of test subjects. Unfortunately, a larger set of test subjects is difficult to obtain due to the actual nature of the SCENT domain. The SCENT domain requires a specific amount of previous training in the areas of recursive programming and LISP or SCHEME syntax. It also assumes that learners are not fully able to solve simple list–recursive problems. This exact amount of training is available for only a short period of time (i.e. 3–4 weeks) in an introductory Computer Science class. Thus the pool of prospective test subjects is very small.

In order to expand this pool, we must choose a domain for which test subjects with the required amount of training are readily available. The best situation would have these test subjects pre-trained; that is, we do not have to perform any of the training ourselves. Using this new domain, we can use the associated larger test subject pool to conduct more accurate and revealing studies into the effectiveness of the PETAL–SCENT style of symbiotic advising systems.

Further to this goal, it is desirable to have some feel for what is involved in devloping a new domain from scratch. The SCENT domain was developed over many years by many different developers and there is no clear understanding as to the effort involved or the pitfalls encountered. A new, scratch domain will provide some insight into what kind of effort is required to develop a new granularity hierarchy, and what common problems may be encountered.

In his thesis, Coulman combines case–based reasoning with the granularity approach to form a powerful diagnosis tool.[4] This combination is realized in the AROMA shell, a graphical user interface that facilitates the construction of granularity hierarchies and case libraries. Using this tool allows a knowledge engineer to design, implement, and test a granularity hierarchy for a particular domain. This tool also allows the task engineer to quickly develop case libraries for a given task. AROMA will be the implementation tool utilized in this study.

3 Trigonometric Proofs

Trigonometric identity proofs provides to us a domain that has a much larger pool of prospective test subjects. Many people have had training in high school–level trigonometry, but their training has not gone beyond that level. This is exactly the amount of training required for this domain.

This domain is also sufficiently formal to use a PETAL–like front end for scaffolding. A standardized set of transformations may be developed that allows the construction of a number of different proofs for a given problem statement. For the scope of this project, I have defined a set of 26 standard transformations; these can be found in Appendix A. A PETAL–like front–end would assist the user by identifying which of this set of transformations were valid and allowing them to alter the problem statement in an attempt to find a proof. The use of this scaffolding front–end ensures that no badly–formed input enters into the granularity hierarchy.

3.1 Transformations in Trigonometric Identity Proofs

A typical trigonometric proof begins with a statement of what is to be proved. For example, the statement,

$$\frac{\csc\beta}{1+\sec\beta} = \frac{\cot\beta}{1+\cos\beta}$$

is a typical problem statement for a proof. The student then makes a series of valid one–step changes to either the left or right side until both sides of the equation are equal. The proof is then complete. Knowing on which side a transformation is performed is not important — the series of transformations is reduceable to a common series. This can be shown best through example. Suppose we are given the problem to prove the equality of:

$$\sin\beta\cot\beta = \cos\beta$$

There are three possible methods to arrive at the proof: changing only the left side, changing only the right side, or changing both the left and right sides. Using the transformation numbers defined in the

Appendix A, these three methods are illustrated following.

1. Left side only.

$$\begin{aligned}
\sin\beta\cot\beta &= \cos\beta \\
\boxed{2}\Downarrow \\
\sin\beta\frac{1}{\tan\beta} &= \cos\beta \\
\boxed{1}\Downarrow \\
\sin\beta\frac{1}{\frac{\sin\beta}{\cos\beta}} &= \cos\beta \\
\boxed{11}\Downarrow \\
\sin\beta\frac{\cos\beta}{\sin\beta} &= \cos\beta \\
\boxed{16}\Downarrow \\
\cos\beta &= \cos\beta
\end{aligned}$$

2. Right side only.

$$\begin{aligned}
\sin\beta\cot\beta &= \cos\beta \\
&\quad \boxed{16}\Downarrow \\
\sin\beta\cot\beta &= \sin\beta\frac{\cos\beta}{\sin\beta} \\
&\quad \boxed{11}\Downarrow \\
\sin\beta\cot\beta &= \sin\beta\frac{1}{\frac{\sin\beta}{\cos\beta}} \\
&\quad \boxed{1}\Downarrow \\
\sin\beta\cot\beta &= \sin\beta\frac{1}{\tan\beta} \\
&\quad \boxed{2}\Downarrow \\
\sin\beta\cot\beta &= \sin\beta\cot\beta
\end{aligned}$$

3. Combination

$$\begin{aligned}
\sin\beta\cot\beta &= \cos\beta \\
\boxed{2}\Downarrow \\
\sin\beta\frac{1}{\tan\beta} &= \cos\beta \\
&\quad \boxed{16}\Downarrow \\
\sin\beta\frac{1}{\tan\beta} &= \sin\beta\frac{\cos\beta}{\sin\beta} \\
\boxed{1}\Downarrow \\
\sin\beta\frac{1}{\frac{\sin\beta}{\cos\beta}} &= \sin\beta\frac{\cos\beta}{\sin\beta} \\
&\quad \boxed{11}\Downarrow \\
\sin\beta\frac{1}{\frac{\sin\beta}{\cos\beta}} &= \sin\beta\frac{1}{\frac{\sin\beta}{\cos\beta}}
\end{aligned}$$

It is important to notice that these proofs are exactly equivalent. Simply reordering the transformations by placing all right side transformations at the end of the sequence results in a common sequence. For example, given the above right side transformation: $16_R 11_R 1_R 2_R$, we can translate this into the equivalent left side transformation simply by reversing the order: $2_R 1_R 11_R 16_R$. (Note that the subscripts are used only in identifying the original transformation position.) Given the combination transformation: $2_L 16_R 1_L 11_R$, we can translate this into the equivalent left side transformation by moving the right transformations to their equivalent positions from the right side of the list. For example, the first right transformation goes in the first position from the right of the list, i.e. the last position in the list. The second right transformation goes in the second position from the right of the list. This results in the list: $2_L 1_L 11_R 16_R$ which is exactly the left side sequence. In this manner, left side and right side transformations become irrelevant; both may be reduced to a common sequence of transformations.

This reduction may only be performed on complete solutions, that is, proofs in which the left and right side are equal in the last step. Frequently, a complete proof is not available, especially when dealing with novice learners. Given a partial proof, a PETAL–like front–end can do a partial ordering, placing a separator character (e.g. a dash) at the position where steps are missing. For example, if the above left–side proof were missing the last step, the sequence $2_L 1_L 11_L -$ would result. If the above combination proof were missing its last two steps, the sequence $2_L - 16_R$ would result. The recognition engine could be altered to perform left side matching up to the separator character first then right side matching up to the separator character when selecting a matching case. In the scope of this paper, all tested transformation sequences were partial left–side transformations (i.e. any missing information was missing from the right side of the sequence).

3.2 Proof Strategies

Common groupings of transformations may be placed into a "proof strategy." This strategy simply groups a set of smaller, constituent unit transformations and/or other proof stategy objects into a proof strategy object. For example, expressing $\cot\beta$ in terms of $\sin\beta$ and $\cos\beta$ involves three unit transformations:

$$\cot\beta \overset{\boxed{2}}{\Rightarrow} \frac{1}{\tan\beta} \overset{\boxed{1}}{\Rightarrow} \frac{1}{\frac{\sin\beta}{\cos\beta}} \overset{\boxed{11}}{\Rightarrow} \frac{\cos\beta}{\sin\beta}$$

When these three transformations are seen within a complete proof, they may be used to identify that the student has changed $\cot\beta$ into $\sin\beta$ and $\cos\beta$ — a higher level proof strategy.

3.3 Capturing Proofs within a Granularity Hierarchy

Using increasing levels of these proof strategies along with the primitive transformations allows us to build up a granularity hierarchy in the trigonometric proof domain. The observers see only the primitive transformations. These observations combine through the higher level proof strategies. All of these objects aggregate together to form a proof.

For a given problem statement, there may be multiple different optimal (or sub–optimal) methods for formulating a proof. Cases may be built that capture these different methods of proving the same problem statement. These cases may be aggregated into a case libarary for a particular task.

These concepts will be illustrated through an example. Consider the problem statement:

$$\frac{\csc\beta}{1+\sec\beta} = \frac{\cot\beta}{1+\cos\beta}$$

There are many different approaches a student may take in a proof. Two are shown in Appendix B: a "straightforward" approach and a "circular" approach. The "straightforward" approach is an example of an optimal proof strategy. The transformations are performed one step at a time from left to right until a solution is reached. Most novice learners will not use this approach since it generally requires understanding the entire proof before starting. This approach yields the transformation string:

$$3 \quad 23 \quad 4 \quad 21 \quad 24 \quad 25 \quad 24 \quad 1 \quad 16 \quad 1 \quad 24 \quad 21 \quad 23 \quad 2 \quad 8$$

It is difficult to pull any higher level proof strategies from this kind of non–novice proof since the transformations were simply applied left to right. Therefore the granularity hierarchy for a proof such as this is left as primitive transformations.

The "circular" approach more accurately resembles a novice approach. Some redundant steps (e.g. application of the circle identity) are applied in an attempt to get to a solution. This approach yields the following transformation string:

$$3 \quad 23 \quad \boxed{5} \quad 21 \quad 21 \quad 4 \quad 25 \quad 24 \quad 1 \quad \boxed{26\,\boxed{5}} \quad 22 \quad 24 \quad 15 \quad 18 \quad 19 \quad 16 \quad 1 \quad 24 \quad 21 \quad 23 \quad 2 \quad 8$$

Notice the two "5" transformations (i.e. the circle identity) along with the higher level "26—5" proof strategy (i.e. expressing $\cos^2\beta$ as $1 - \sin^2\beta$). These items (along with others) serve to distiguish the second proof strategy from the first. That is, they provide it with a unique "finger–print."

The above proofs (possibly along with more) may be included in the case library. A student's proof will match most closely with one of these stored proofs and advice may be given to assist the student through a difficulty.

4 The Case Study

The first step in developing a granularity hierarchy for a domain is to find the primitive elements in that domain. In trigonometric identity proofs, the primitive item was a transformation – a basic, single algebraic or trigonometric step. Thus, a proof is simply a series of transformations. A valid proof may alter both the left and right side of the equation, however this is not important with the representation here since the sequence of transformations may undergo simple reordering to fall into a common formulation. A PETAL–like scaffolding front–end can handle the translation from mathematical representations to a simple transformation sequence used by the granularity hierarchy recognition engine. It can also ensure that no illegal transformations are made. This saves the knowledge engineer from having to design the granularity hierarchy with this in mind.

The number of primitive elements to use is an important decision in developing a granularity hierarchy. A scaffolding front–end can be used as an initial filter to remove some of the unwanted "lint" from a student solution before it is sent to the recognition engine. Stated another way, some of the primitive elements can be considered to be "useful" in determining user strategies, while others can be filtered out at the front–end. There is a trade–off involved in this decision step — the more primitives available to the granularity hiararchy, the more uniquely a solution can be identified. However, increased numbers of primitives increases the amount of computational complexity and therefore increases response time. It is important to remove the unnecessary primitives in order to achieve a small, robust granularity hierarchy.

In the trigonometric identity proof domain, all 26 primitives from Appendix A were used in the granularity hierarchy. Although some of them were not "useful," their numbers were not large enough to incur computational penalty. Also, a scaffolding front–end may indeed provide many more transformations than these (due to the mutability of algebra); in that case, only these 26 need be considered "useful," all others may be filtered out.

After the primitive elements have been identified and it has been verified that the domain can indeed be represented completely using these elements, the knowledge engineer must then attempt to combine the unit elements into higher level, possibly abstract, strategy objects. This combination

High Level Strategy	Transformations
$\cos^2\beta \rightarrow 1-\sin^2\beta$	$26-5$
$\sin^2\beta \rightarrow 1-\cos^2\beta$	$26-5$
$\cot\beta \rightarrow \frac{\cos\beta}{\sin\beta}$	$2-1-11$
$\tan^2\beta \rightarrow \frac{\sin^2\beta}{\cos^2\beta}$	$1-12$

Table 1: High level strategy objects within the trigonometric identity proof domain.

should continue to as high and as abstract a level as possible, provided that they stay within the domain and computational feasibility (that is, "useful"). The more distinguishing marks in the finger–print, the easier it is to uniquely identify and comment on it.

Within the trigonometry domain, this combination was a bit tricky. An attempt was made to combine primitives in a manner that would indicate that a student was using a particular proof technique. For example, it is a two step process to express $\tan^2\beta$ in terms of $\sin\beta$ and $\cos\beta$. These two steps were combined into a higher level proof strategy to indicate when a student has used these two steps in conjuction to accomplish this technique. This is an example of a "useful" combination. There are $n!$ different combinations of the transformations, however, most serve no purpose in identifying student stategies. This is an important consideration in deciding which primitives to include in the granularity hierarchy. Table 1 describes the higher level strategies currently recognized in the trigonometric identity proof domain.

Finally, something must be said about advice quality. As more connections are discovered between primitives and the granularity hierarchy and case library are subsequently expanded, the ability to give quality advice is increased. The granularity hierarchy is able to give advice on more and more abstract concepts at higher and higher levels. Thus, it is important to include as many primitives and discover as many connections as is possible — the quality of advice depends on it. The case library should try and cover the abstract strategy objects. That is, the task engineer should develop a set of cases (optimal, sub–optimal or incorrect) that covers all the abstract strategy objects within the granularity hierarchy. In doing this, the task engineer ensures that there is a relevant case for each strategy recognized by the granularity hierarchy. In the trigonometric identity proof domain, according to this rule, each task requires four cases to cover the set of defined strategy objects (see Table 1).

5 Conclusions

The effectiveness of the symbiotic, SCENT–PETAL–like, advising systems need to be qualitatively measured. Existing studies have indicated that these systems may indeed be useful, but have been woefully inadequate in terms of numbers of test subjects. This is due simply to the nature of the SCENT domain itself: recursive LISP or SCHEME functions. A larger test pool is required to more accurately measure effectiveness.

The proof of trigonometric identities is an area in which many people have had training. Many students have taken training in this area up to the level of Grade Twelve Trigonometry but not beyond. This is the perfect amount of training required for this domain. The domain itself is very well defined and mechanical enough to allow the use of a PETAL–like front–end to assist the student in making valid transformations and ensure the absence of erroneous input. It can also be used to filter out unwanted information.

The trigonometric primitives can be combined into higher level strategy objects to uniquely identify many different proof techniques. Given a sufficient number of abstract proof strategies, useful advice may be tacked onto a student solution. The power of the advice is limited only by the extent to which the knowledge engineer combines proof strategies and the number of example proofs in the case library. The difficulties involved in determining primitives for a domain and the extent to which the granularity hierarchy and the case library should be expanded have been discussed.

Given a complete granularity hierarchy, fully augmented with advice, and an extensive case library, the completion of a comprehensive study concerning the effectiveness of a symbiotic PETAL–SCENT–like system is possible.

References

[1] Anderson, J. R., and Reiser, B. J. The LISP tutor. *Byte* (Apr. 1985), 159–175.

[2] Bhuiyan, S. *Identifying and supporting mental models of recursion in a learning environment.* PhD thesis, University of Saskatchewan, 1992.

[3] Collins, J. A. PETAL–SCENT advice experiment. Computer Science 880 Miniproject 2, 1994.

[4] Coulman, R. A. Combining case–based reasoning and granularity for educational diagnosis in an intelligent tutoring system. Master's thesis, University of Saskatchewan, 1991.

[5] Coulman, R. A., Greer, J. E., and McCalla, G. I. Augmenting granularity–based recognition for the real world: sailing the four C's. Tech. rep., ARIES Laboratory 94-3, University of Saskatchewan, 1994.

[6] McCalla, G., Greer, J., Barrie, B., and Pospisil, P. Granularity hierarchies. In *Semantic Networks in Artificial Intelligence*, F. Lehmann, Ed. Pergamon Press, Ltd., Oxford, 1992, pp. 363–375.

[7] McCalla, G. I., and Greer, J. E. Granularity–based reasoning and belief revision in student models. In *Student Modelling: The Key to Individualized Knowledge–Based Instruction*, J. E. Greer and G. I. McCalla, Eds. Springer–Verlag, Berlin, 1994, pp. 39–62.

[8] McCalla, G. I., and Greer, J. E. Learning about programming. In *Emerging Computer Technologies in Education*, T.-W. Chan and J. Self, Eds. AACE Press, Virgina, U.S.A., to appear.

[9] Price, B. R., McCalla, G. I., and Greer, J. E. Combining scaffolding and diagnosis in a distributed tutoring system. In *East–West Conference on Computer Technologies in Education EW–ED '94* (1994).

A Trigonometric and Algebraic Transformations

$$\frac{\sin\beta}{\cos\beta} \overset{\boxed{1}}{=} \tan\beta$$
$$\frac{1}{\tan\beta} \overset{\boxed{2}}{=} \cot\beta$$
$$\frac{1}{\sin\beta} \overset{\boxed{3}}{=} \csc\beta$$
$$\frac{1}{\cos\beta} \overset{\boxed{4}}{=} \sec\beta$$
$$1 \overset{\boxed{5}}{=} \sin^2\beta + \cos^2\beta$$
$$\frac{x+y}{z} \overset{\boxed{6}}{=} \frac{x}{z} + \frac{y}{z}$$
$$\frac{x-y}{z} \overset{\boxed{7}}{=} \frac{x}{z} - \frac{y}{z}$$
$$x+y \overset{\boxed{8}}{=} y+x$$
$$xy \overset{\boxed{9}}{=} yx$$
$$\frac{x}{x} \overset{\boxed{10}}{=} 1$$
$$\frac{x}{y} \overset{\boxed{11}}{=} \frac{1}{\frac{y}{x}}$$
$$\frac{x^2}{y^2} \overset{\boxed{12}}{=} \left(\frac{x}{y}\right)^2$$
$$x^2y^2 \overset{\boxed{13}}{=} (xy)^2$$
$$aa \overset{\boxed{14}}{=} a^2$$
$$1 \overset{\boxed{15}}{=} 1^2$$
$$y \overset{\boxed{16}}{=} x\left(\frac{y}{x}\right)$$
$$(1+x)(1-x) \overset{\boxed{17}}{=} 1-x^2$$
$$x-x \overset{\boxed{18}}{=} 0$$
$$0+x \overset{\boxed{19}}{=} x$$
$$0\times x \overset{\boxed{20}}{=} 0$$
$$x(y+z) \overset{\boxed{21}}{=} (xy)+(xz)$$
$$x(y-z) \overset{\boxed{22}}{=} (xy)-(xz)$$
$$\frac{\frac{1}{x}}{y} \overset{\boxed{23}}{=} \frac{1}{xy}$$
$$1\times x \overset{\boxed{24}}{=} x$$
$$x\left(\frac{y}{z}\right) \overset{\boxed{25}}{=} \frac{xy}{z}$$
$$x \overset{\boxed{26}}{=} x+y-y$$

B Example Proof Derivations

This section describes two proofs for the problem statement:

$$\frac{\csc\beta}{1+\sec\beta} = \frac{\cot\beta}{1+\cos\beta}$$

The first subsection derives an straightforward, optimal proof, while the second subsection shows a more indirect method that uses the circle identity in the derivation.

- Straightforward Optimal Proof

$$
\begin{aligned}
\frac{\csc\beta}{1+\sec\beta} &\overset{\boxed{3}}{\Rightarrow} \frac{\frac{1}{\sin\beta}}{1+\sec\beta}\\
&\overset{\boxed{23}}{\Rightarrow} \frac{1}{\sin\beta\,(1+\sec\beta)}\\
&\overset{\boxed{4}}{\Rightarrow} \frac{1}{\sin\beta\left(1+\frac{1}{\cos\beta}\right)}\\
&\overset{\boxed{21}}{\Rightarrow} \frac{1}{\sin\beta\times 1+\sin\beta\times\frac{1}{\cos\beta}}\\
&\overset{\boxed{24}}{\Rightarrow} \frac{1}{\sin\beta+\sin\beta\times\frac{1}{\cos\beta}}\\
&\overset{\boxed{25}}{\Rightarrow} \frac{1}{\sin\beta+\frac{\sin\beta\times 1}{\cos\beta}}\\
&\overset{\boxed{24}}{\Rightarrow} \frac{1}{\sin\beta+\frac{\sin\beta}{\cos\beta}}\\
&\overset{\boxed{1}}{\Rightarrow} \frac{1}{\sin\beta+\tan\beta}\\
&\overset{\boxed{16}}{\Rightarrow} \frac{1}{\frac{\sin\beta}{\cos\beta}\times\cos\beta+\tan\beta}\\
&\overset{\boxed{1}}{\Rightarrow} \frac{1}{\tan\beta\cos\beta+\tan\beta}\\
&\overset{\boxed{24}}{\Rightarrow} \frac{1}{\tan\beta\cos\beta+\tan\beta\times 1}\\
&\overset{\boxed{21}}{\Rightarrow} \frac{1}{\tan\beta\,(\cos\beta+1)}
\end{aligned}
$$

$$\overset{\boxed{23}}{\Longrightarrow} \frac{\frac{1}{\tan\beta}}{\cos\beta+1}$$

$$\overset{\boxed{2}}{\Longrightarrow} \frac{\cot\beta}{\cos\beta+1}$$

$$\overset{\boxed{8}}{\Longrightarrow} \frac{\cot\beta}{1+\cos\beta}$$

- Proof Using Circle Identity

$$\frac{\csc\beta}{1+\sec\beta} \overset{\boxed{3}}{\Longrightarrow} \frac{\frac{1}{\sin\beta}}{1+\sec\beta}$$

$$\overset{\boxed{23}}{\Longrightarrow} \frac{1}{\sin\beta\,(1+\sec\beta)}$$

$$\overset{\boxed{5}}{\Longrightarrow} \frac{1}{\sin\beta\left(\sin^2\beta+\cos^2\beta+\sec\beta\right)}$$

$$\overset{\boxed{21}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\,(\cos^2\beta+\sec\beta)}$$

$$\overset{\boxed{21}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\cos^2\beta+\sin\beta\sec\beta}$$

$$\overset{\boxed{4}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\cos^2\beta+\sin\beta\left(\frac{1}{\cos\beta}\right)}$$

$$\overset{\boxed{25}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\cos^2\beta+\frac{\sin\beta\times 1}{\cos\beta}}$$

$$\overset{\boxed{24}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\cos^2\beta+\frac{\sin\beta}{\cos\beta}}$$

$$\overset{\boxed{1}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\cos^2\beta+\tan\beta}$$

$$\overset{\boxed{26}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\left(\cos^2\beta+\sin^2\beta-\sin^2\beta\right)+\tan\beta}$$

$$\overset{\boxed{5}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\left(1-\sin^2\beta\right)+\tan\beta}$$

$$\overset{\boxed{22}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta\times 1-\sin\beta\sin^2\beta+\tan\beta}$$

$$\overset{\boxed{24}}{\Longrightarrow} \frac{1}{\sin\beta\sin^2\beta+\sin\beta-\sin\beta\sin^2\beta+\tan\beta}$$

$$\overset{\boxed{15}}{\Longrightarrow} \frac{1}{\sin\beta+\sin\beta\sin^2\beta-\sin\beta\sin^2\beta+\tan\beta}$$

$$\overset{\boxed{18}}{\Longrightarrow} \frac{1}{\sin\beta+0+\tan\beta}$$

$$\overset{\boxed{19}}{\Longrightarrow} \frac{1}{\sin\beta+\tan\beta}$$

$$\overset{\boxed{16}}{\Longrightarrow} \frac{1}{\frac{\sin\beta}{\cos\beta}\times\cos\beta+\tan\beta}$$

$$\overset{\boxed{1}}{\Longrightarrow} \frac{1}{\tan\beta\cos\beta+\tan\beta}$$

$$\overset{\boxed{24}}{\Longrightarrow} \frac{1}{\tan\beta\cos\beta+\tan\beta\times 1}$$

$$\overset{\boxed{21}}{\Longrightarrow} \frac{1}{\tan\beta\,(\cos\beta+1)}$$

$$\overset{\boxed{23}}{\Longrightarrow} \frac{\frac{1}{\tan\beta}}{\cos\beta+1}$$

$$\overset{\boxed{2}}{\Longrightarrow} \frac{\cot\beta}{\cos\beta+1}$$

$$\overset{\boxed{8}}{\Longrightarrow} \frac{\cot\beta}{1+\cos\beta}$$