RODRIGO MORANDE BECKER

# PROGREF V4:

## UM *SOFTWARE* PARA COLETA DE DADOS EM PROGRAMAS DE REFORÇO COM HUMANOS

Londrina
2011

RODRIGO MORANDE BECKER

# PROGREF V4:

## UM *SOFTWARE* PARA COLETA DE DADOS EM PROGRAMAS DE REFORÇO COM HUMANOS

Dissertação apresentada ao programa de Pós-Graduação em Análise do Comportamento da Universidade Estadual de Londrina, como requisito parcial à obtenção do título de Mestre em Análise do Comportamento.

Orientador: Prof. Dr. Carlos Eduardo Costa

Londrina
2011

**Catalogação elaborada pela Divisão de Processos Técnicos da Biblioteca Central da Universidade Estadual de Londrina.**

**Dados Internacionais de Catalogação-na-Publicação (CIP)**

B395p Becker, Rodrigo Morande.
ProgRef v4 : um software para coleta de dados em programas de reforço com humanos / Rodrigo Morande Becker. – Londrina, 2011.
103 f. : il.

Orientador: Carlos Eduardo Costa.
Dissertação (Mestrado em Análise do Comportamento) – Universidade Estadual de Londrina, Centro de Ciências Biológicas, Programa de Pós-Graduação em Análise do Comportamento, 2011.
Inclui bibliografia.

1. Reforço (Psicologia) – Teses. 2. Comportamento humano – Pesquisa experimental – Tecnologia – Teses. 3. Psicologia – Software de aplicação – Teses. 4. Informática na psicologia – Teses. I. Costa, Carlos Eduardo. I. Universidade Estadual de Londrina. Centro de Ciências Biológicas. Programa de Pós-Graduação em Análise do Comportamento. III. Título.

CDU 159.9.019.43:519.681

RODRIGO MORANDE BECKER

# PROGREF V4:

## UM *SOFTWARE* PARA COLETA DE DADOS EM PROGRAMAS DE REFORÇO COM HUMANOS

Dissertação apresentada ao programa de Pós-Graduação em Análise do Comportamento da Universidade Estadual de Londrina, como requisito parcial à obtenção do título de Mestre em Análise do Comportamento

**BANCA EXAMINADORA**

_____

Prof. Dr. Marcelo Frota Lobato Benvenuti
UNB - Brasilia - GO

_____

Prof. Dr. Célio Roberto Estanislau
UEL – Londrina - PR

_____

Prof. Dr. Carlos Eduardo Costa
UEL – Londrina - PR

Londrina, 11 de agosto de 2011.

BECKER, R. M. **ProgRef v4**: um *software* para coleta de dados em programas de reforço com humanos. 2011. 103 f. Dissertação (Mestrado em Análise do Comportamento) - Universidade Estadual de Londrina, Londrina, PR, 2011.

## RESUMO

Os avanços científico e tecnológico estão cada vez mais interdependentes. A Análise Experimental do Comportamento como uma ciência não pode se colocar distante de meios que facilitem e aprimorem a coleta e a análise de dados. Um computador de uso pessoal é uma ótima ferramenta para auxiliar na pesquisa tanto básica quanto aplicada. A presente dissertação esta inserida nesse contexto ao propor o desenvolvimento de um *software* para coleta e análise de dados de programas de reforço com humanos. Visando uma introdução aos programas de reforço como um arranjo experimental para o estudo da seleção operante, o Capítulo 1 define alguns conceitos chaves e propõe a discussão sobre a utilidade em se utilizar programas de reforço. As diversas áreas da ciência se beneficiaram da evolução tecnológica propiciada pela “era digital”. O Capítulo 2 discute como foi esse desenvolvimento com um breve histórico e apresenta alguns exemplos bem sucedidos dessa interacao tecnologia-pesquisa, dando enfase na utilização das tecnologias pela Psicologia. O novo *software*, qui apresentado, é uma atualização do ProgRef v3. É realizada uma análise crítica do *software* ProgRefv3 avaliando tanto questões técnicas quanto produtividade academica: pesquisas desenvolvidas e os resultados atingidos. Por fim, é apresentado o ProgRefv4, seguindo o objetivo de introduzir conceitos de áreas tecnológica, principalmente da informática, à estudantes e pesquisadores da Análise Expeimental do Comportamento, é relacionado alguns pontos técnicos do desenvolvimento do novo *software.* Seguido de uma introdução à utilizacao dos programas ProgRef v4, para coleta de dados e ProgRef_DA, para análise dos dados. É realizada uma descrição dos principais recursos disponíveis no *software*, servindo como um manual introdutório para os pesquisadores.

**Palavras-chave**: Esquemas de reforçamento. Tecnologia. S*oftware* para pesquisa. Análise experimental do comportamento. ProgRef.

BECKER, R. M. **ProgRef v4**: a software for data collection on schedules of reinforcement with humans. 2011. 103 f.Dissertation presented to the Pos-Graduation Program in Behaior Analysis of Londrina State University, as a partial requisite for the fullfilment of the Master in Behavior Analysis title from Londrina State University, Londrina,PR, 2011.

## ABSTRACT

Scientific and technological advances are each time more interdependent. The Experimental Analysis of Behavior as a science cannot stand apart from means to facilitate and improve the collection and analysis of data. A personal computer is a great tool to aid both basic and applied research. The current dissertation fits this context proposing the development of a software for data collection and analysis on schedules of reinforcement with humans. Chapter 1 is an introduction of the schedules of reinforcement as an experimental arrangement for the study of operant selection. It defines some key concepts and proposes a discussion about the convenience of using the schedules of reinforcement. Science has benefited from technological changes brought about by the "digital age". Chapter 2 discusses this development presenting a brief history and some successful examples of this technology-research interaction, the use of technology by psychology is highlighted. The new software, here presented, is an update of ProgRef v3. First, it's performed a critical analysis of this software evaluating both technical aspects and academic productivity: research conducted and results achieved. Next, ProgRefv4 is presented complementing the introduction of concepts in the areas of technology, especially computing sciences. To students and researchers of the Experimental Analysis of Behavior, some technical issues of the development of the new software are listed. Subsequently, there is an introduction to the use of the programs: ProgRef v4 for data collection, and ProgRef_DA for data analysis. Lastly, there's a description of the main features available in the software, serving as an introductory manual for researchers.

**Key-words**: Schedules of reinforcement. Technology. Software for research. Experimental analysis of behavior. ProgRef.

## LISTA DE ILUSTRAÇÕES

## LISTA DE ABREVIATURAS E SIGLAS

AEC - Análise Experimental do Comportamento
CAS - *Computer Attitude Scale*
CERN - *Conseil Européen pour la Recherche Nucléaire* (em francês)
CLR - *Common Language Runtime*
CRF - Reforço Contínuo
DRL - Reforço Diferencial de Baixa Taxa
DRH - Reforço Diferencial de Alta Taxa
ENIAC - Electronic Numerical Integrator and Computer
EXT - Extinção
FI - Intervalo Fixo
FR - Razão Fixa
FT - Tempo Fixo
HD - *Hard Disk*
IA - Inteligência Artificial
IBGE - Instituto Brasileiro Geografia e Estatística
IP - *Internet Protocol*
IRI - Intervalo entre Reforços
IRT - intervalo entre respostas
JEAB - *Journal of the Experimental Analysis of Behavior*
LH - *Limited Hold*
LHC - *Large Hadron Collider*
PDP - Processamento Distribuído em Paralelo
POO - Programação Orientada a Objeto
PP - Programação Procedural
PRP - Pausa Pós-Reforço
RC - Resposta Consumatória
SO - Sistema Operacional
TO - Time Out
VBA - *Visual Basic for Applications*
VI - Intervalo Variável
VR - Razão Variável
VT - Tempo Variável

**SUMÁRIO**

## APRESENTAÇÃO

O avanço tecnológico proporciona às ciências o desenvolvimento de equipamentos e aprimoramento de métodos para que seus objetos de estudo sejam mais bem compreendidos. Lattal (2005; 2008) discute a importância da tecnologia para a Análise Experimental do Comportamento e argumenta que o desenvolvimento tecnológico de outras áreas pode auxiliar e viabilizar pesquisas na Análise do Comportamento. Mesmo nos trabalhos iniciais de Skinner, por exemplo, o desenvolvimento das pesquisas utilizando a pressão à barra pelo rato dependeu de outras áreas como eletrônica, engenharia elétrica, mecânica e fisiologia experimental. Em uma busca histórica, Lattal encontrou que muitos dos aparatos descritos no *Journal of the Experimental Analysis of Behavior* (JEAB) apresentavam algum componente que não foi inicialmente projetado para a Análise Experimental do Comportamento.

A utilização de equipamentos de outras áreas pode ser visto em um dos aparatos mais utilizados pela Análise Experimental do Comportamento, o registrador cumulativo. Ele foi uma adaptação de um equipamento originalmente projetado para a fisiologia. Este equipamento também fora muito utilizado em pesquisas na área da genética, como descrito por Weiner (2001). O equipamento, que inicialmente fora utilizado em estudos de fisiologia,a se mostrou muito produtivo para diversas áreas. Alguns fatores que propiciaram esse uso, provavelmente foram: a sua mecânica simples, apenas um motor com velocidade constante e uma caneta (pena); e uma sensibilidade capaz de registrar pequenos "ruídos", sejam eles gerados por componentes eletrônicos (e.g., a interrupção de um feixe de infravermelho) ou mecânicos (e.g., a chave comutadora de uma caixa de Skinner que servia como operandum). Lattal (2008) utiliza o termo "tecnologia exógena" para indicar as influências tecnológicas de outras campos científicos na Análise do Comportamento.

A computação pode ser considerada uma tecnologia exógena que influencia a Análise Experimental do Comportamento. No entanto, Killeen (1985) criticou a utilização tardia e deficitária dessa tecnologia pelos analistas do comportamento. Com o auxílio de computadores, muitas formas de registros do comportamento tornaram-se possíveis, mas pouco havia sido tentado até aquela época e a utilização do registrador cumulativo permaneceu a principal fonte de coleta de dados. O computador é uma importante ferramenta para coleta e análise de dados, além de outras coisas, como produção de textos, pesquisas bibliográficas etc.

Atualmente é possível a um cientista conectar seu computador pessoal à grande rede e entrar imediatamente em contato com o que há de mais recente na divulgação de informações científicas. Mas esse "simples computador", que possibilitou ao cientista usufruir dos benefícios da Internet, também pode facilitar e melhorar a sua produção acadêmica, não só na busca de conhecimentos ou referências bibliográficas, mas também na coleta e análise de dados. Para isso, basta que ele empregue algumas horas no aprendizado da utilização dos diversos *software*s (comercializados ou colocados à disposição gratuitamente) ou se dedique no aprendizado de uma linguagem de programação, ampliando enormemente o potencial do computador como uma ferramenta de coleta e análise de dados. No entanto, para muitas pessoas, inclusive pesquisadores, a tecnologia não passa de aparelhos "caixa-preta", para não dizer "dispositivos mágicos"[1]. Assim, a interação com esses equipamentos é feita de forma deficitária, pois grande parte dos recursos disponíveis nunca é utilizada.

Através do desenvolvimento de uma cultura tecnológica talvez fosse possível reverter esse déficit, facilitando o desenvolvimento de novas tecnologias e consequentemente o avanço científico de uma área. O termo "cultura tecnológica" é aqui utilizado para referir à interação entre pessoas e o ambiente tecnológico. Esse ambiente é caracterizado por conceitos, aparelhos e instrumentos tecnológicos. No presente trabalho a tecnologia será entendida como as técnicas, conhecimentos, métodos, materiais, ferramentas e processos usados para resolver ou, ao menos, facilitar a solução de problemas. A interação entre pessoa e tecnologia pode ser avaliada em termos de sua produtividade. A produtividade, no âmbito computacional, está muito mais ligada à capacidade de utilização ou entendimento por parte do usuário dos produtos da tecnologia, do que com o ambiente gerado por essas tecnologias em si mesmo. Ou seja, hoje os instrumentos computacionais (tanto hardware quanto *software*) estão facilmente à disposição, tanto do ponto de vista de custo quanto de materiais de estudo. Ao experimentador cabe a aprendizagem no uso dessas tecnologias.

Um bom exemplo do avanço tecnológico impulsionando o avanço científico – e como os pesquisadores de uma área interagem com esses aparatos – pode ser visto na área da Física. O CERN, fundado em 1954, é um dos maiores e mais respeitados centros de pesquisa científica. Nele se encontra o Grande Colisor de Hádrons (LHC – sigla em inglês), que entrou em atividade em setembro de 2008. Ele é um acelerador de partículas usado pelos físicos para o estudo das menores partículas conhecidas – o "bloco de construção" fundamental de todas as coisas. A expectativa é que esse instrumento revolucione nosso

[1] O termo "dispositivos mágicos" é utilizado em uma linguagem figurada para exemplificar que muitas pessoas desconhecem a lógica interna dos sistemas digitais.

entendimento do mundo “muito pequeno” – interior do átomo – e do mundo “muito grande” – da vastidão do universo. Com este equipamento seria possível corroborar ou descartar algumas das teorias mais controversas vigentes na física moderna (Collins, 2008).

Quando se fala em gigantescos equipamentos processando 10 gigabits[2] por segundo e físicos analisando esses mesmos dados fica difícil separar, de um lado, os conhecimentos das diversas engenharias eletrônica, mecânica e de *software* e, de outro lado, os conhecimentos da Física. Esses grupos de cientistas trabalham conjuntamente para atingir os objetivos de uma ciência com princípios bem estabelecidos. Essa integração, tão presente e produtiva entre estudos científicos na Física, pode ter sido possibilitada pela cultura tecnológica dos participantes envolvidos. Físicos têm uma formação voltada para a cultura tecnológica. Na ementa de seus cursos é possível encontrar matérias como linguagem de programação e várias outras voltadas para tecnologia, isso sem mencionar o contato que o aluno tem com instrumentos de interface em seus laboratórios experimentais desde a graduação. Como apontado por Beagley (2001), essa cultura tecnológica ainda se mostra muito precária na Análise Experimental do Comportamento.

Um estudo experimental desenvolvido por Anderson e Hornby (1996) buscou investigar os fatores que poderiam influenciar na atitude positiva ou negativa diante de computadores por alunos do curso de Psicologia. Os participantes eram de quatro universidades que tinham em sua grade curricular o uso de computadores com objetivos distintos. As variáveis utilizadas foram: gênero, idade, ano escolar, lócus de controle e experiência prévia com computadores. O método utilizado foi o uso de questionários e testes padronizados, aplicados pré- e pós-curso. Apesar da limitação na interpretação dos resultados, uma vez que as medidas foram todas indiretas, a variável de maior relevância para uma atitude positiva[3] ante aos computadores foi a experiência prévia com computadores. Outro resultado importante foi que as atitudes dos alunos em relação aos computadores mudaram ao longo do curso, de uma atitude negativa para uma positiva, quando os computadores foram utilizados em seu curso. O ponto que parece importante destacar aqui é que a exposição ao uso de computadores (nos cursos de graduação) pode mudar o comportamento dos alunos (e futuros profissionais) em relação à tecnologia e seus usos.

Muitos pesquisadores analistas do comportamento parecem não fazer questão de aprender uma linguagem de programação, o que poderia melhorar a utilização dos

---

[2] Um *gigabit* é uma unidade de armazenamento de informações ou dados de computadores. Normalmente ele é abreviado por Gb [1 Gb = 1.000.000.000 bits ($10^9$)].

[3] Este termo foi usado pelos autores tendo como referência os testes padronizados, escala de atitude ao computador (CAS, na sigla em inglês), um alto *score* no teste indica uma atitude mais positiva.

computadores. Se dedicar ao conhecimento de novos *software*s ou à aprendizagem de linguagem de programação poderia auxiliar até mesmo no processo de formulação de hipóteses para os possíveis problemas na área de pesquisa. Por exemplo, na área aplicada da Psicologia do Esporte, uma variável dependente poderia ser o tempo de reação na largada em uma prova de 100 metros rasos no atletismo. A medida dessa variável requer instrumentos capazes de registros precisos na casa de milissegundos. Seria impraticável coletar esses dados com cronômetros operados manualmente. Sem os equipamentos e os *software*s adequados para coleta e análise dos dados, o experimentador ficaria limitado ao investigar esse fenômeno. Uma alternativa seria contratar profissionais da área tecnológica para desenvolver os equipamentos e os *software*s necessários para a pesquisa, mas o desconhecimento da área tecnológica também dificulta a comunicação com esses profissionais. Beagley (2001) discute que tal posicionamento leva o experimentador a olhar de forma simplista para a tecnologia que precisa ser desenvolvida, muitas vezes influenciado pela interface amigável dos *software*s desenvolvidos. O autor prossegue argumentando que, por sua vez, os profissionais com formação em outros campos científicos também olham para processos comportamentais de forma simplista, muitas vezes influenciados por leituras equivocadas da bibliografia pertinente. Todo esse descompasso entre o pesquisador e o desenvolvedor só trás resultados perniciosos ao produto final – o *software* para a pesquisa.

Os fatores para tal descaso provavelmente são vários, mas pode-se especular que a falta de cultura tecnológica – gerada, entre outras coisas, por currículos acadêmicos que não modelam comportamentos "pró-tecnologia" – na formação em Psicologia em geral e na Análise do Comportamento em particular, contribui muito para este cenário. Isto poderia ser sanado, por exemplo, com maiores incentivos para publicações de artigos técnicos envolvendo desenvolvimento de tecnologias. Uma enorme contribuição para isso é o periódico *Behavior Research Methods*[4], que trás diversos artigos envolvendo tecnologia na pesquisa em Psicologia. Também é possível ver esforços nesse sentido na "Revista Brasileira de Terapia Comportamental e Cognitiva", que coloca à disposição a seção "Nota Técnica", para a divulgação de trabalhos envolvendo soluções em *software* ou hardware para a Psicologia (ver, por exemplo, Costa & Banaco, 2002, 2003; Debert & Andery, 2005; Pessoa & Buffara, 2005). Outra forma de difundir conhecimentos técnicos e o uso de tecnologia seria

[4] *Behavior Resarch Methods* (ver sítio: http://brm.psychonomic-journals.org/) é o nome do periódico a partir do ano de 2001. Para os números anteriores a essa data o nome do periódico era *Behavior Research Methods, Instruments & Computers* (ver sítio: www.psychonomic.org/).

proporcionar o acesso à tecnologia desde a graduação nos cursos de Psicologia, como sugerem os resultados da pesquisa de Anderson e Hornby (1996), citada anteriormente.

Além da postura dos cientistas quanto à tecnologia, outro fator que dificulta a utilização de equipamentos é a forma de seu financiamento, que geralmente, tem um custo elevado. Como discutido por Braden (1996), os recursos financeiros para pesquisas científicas sofrem oscilações que são reflexos da situação político-econômica de um país e das ideologias de seus governantes. Áreas que desenvolvem pesquisas que possam ser lucrativas no futuro têm à disposição um grande incentivo do capital privado, como por exemplo, pesquisas na área farmacêutica. Mas esse não é o caso da maior parte das pesquisas em Análise Experimental do Comportamento que, no Brasil, depende de recursos de órgãos governamentais de incentivo à pesquisa e divulgação científica, que muitas vezes são escassos ou não atendem adequadamente a área.

Superadas tais dificuldades, ou seja, após o cientista compreender o avanço tecnológico e saber utilizar dos benefícios que ele proporciona e obter os recursos necessários para aquisição dos equipamentos por meio de órgãos de fomento à pesquisa, ele sairá em busca de equipamentos comerciais. No entanto, nessa busca, outras dificuldades são encontradas. Os equipamentos comerciais trazem alguns problemas devido a sua origem em arranjos mercadológicos (oferta vs. demanda). Beagley (2001) argumenta que a venda de equipamentos para pesquisa não é uma área lucrativa e, portanto, as grandes empresas que possuem expertise tecnológica e pessoas altamente qualificadas não entram nesse nicho de mercado. Empresas de pequeno porte se aventuram nesse ramo. Isso pode trazer problemas nos estudos científicos, pois as pequenas empresas não possuem uma estrutura de pós-venda e atualizações adequadas como as grandes empresas. Com isso, a correção de possíveis bugs ou ajustes dos equipamentos podem sofrer grandes atrasos, comprometendo o cronograma da pesquisa, senão, a pesquisa como um todo.

Como uma tentativa para se manter no mercado e obter lucros com o desenvolvimento dos equipamentos ou *software*s, as empresas procuram desenvolver produtos generalistas, isto é, que abrangem um amplo ramo de pesquisas e possibilitam diversos arranjos experimentais. Esse tipo de foco pode parecer benéfico no princípio, mas logo se revela limitado. Equipamentos generalistas dificultam e muitas vezes até restringem sua utilização em problemas experimentais específicos, pois, por mais abrangente que possa ser o equipamento ou *software*, nunca será capaz de acompanhar a criatividade do pesquisador, o interesse por novas variáveis e novos arranjos metodológicos.

Esses *software*s generalistas obrigam muitas vezes o experimentador a aprender a linguagem do equipamento, como por exemplo, o MED-PC e o SKED-II[5] (Gollub, 1991). Essa programação pode ser excessivamente trabalhosa, e a relação custo-benefício deve ser bem analisada. Para melhorar essa relação o pesquisador poderia aprender uma linguagem de programação de alto-nível[6].

Aos cientistas interessados em aprender uma linguagem de programação e utilizá-la para desenvolver *software*s para a realização de suas pesquisas, Gollub (1991) faz uma análise crítica das vantagens e desvantagens no uso de uma linguagem de programação de alto-nível por parte dos pesquisadores em Psicologia (em vez de optar pela utilização de uma linguagem restrita ao equipamento, como as citadas anteriormente). Uma descrição dessas vantagens e desvantagens será realizada no Capítulo 2.

Jansen, Wiertz, Meyer e Noldus (2003) discutiram os problemas e os benefícios que são trazidos a um determinado estudo pela maneira como um *software* é construído (i.e., sua lógica da programação, layout dos componentes na tela, dados coletados etc.). Esses autores estavam interessados em garantir um bom tratamento e consistência nos dados de observações comportamentais. Para isso analisaram e descartaram vários *software*s que apresentavam diversos problemas em sua construção. Com os dados levantados, desenvolveram o *software* "The Observer 4.1" que, através de uma programação mais adequada, atingiu o grau de confiabilidade desejada e atingiu os objetivos, tantos os levantados pela busca bibliográfica, quanto novos para um *software* de observação comportamental.

Uma vez que os recursos financeiros para pesquisa, de um modo geral, são escassos e soluções comerciais que atendam os interesses da pesquisa sobre programas de reforço[7] com humanos em Análise Experimental do Comportamento são praticamente

---

[5] Uma descrição e análise do *software* MED-PC será realizada no Capítulo 2. Para o leitor interessado segue *site* do fabricante: MED-PC: www.med-associates.com/*software*/medpc.htm (acessado em 04/01/2010).

[6] Uma linguagem de programação é um método padronizado para expressar instruções para um computador. É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador. Uma linguagem permite que um programador especifique precisamente sobre quais dados um computador vai atuar, como estes dados serão armazenados ou transmitidos e quais ações devem ser tomadas sob várias circunstâncias. São divididas em linguagem de alto-nível, composta de símbolos mais complexos, inteligível pelo ser humano e não-executável diretamente pela máquina no nível da especificação de algoritmos (e.g., Basic, Fortran, C/C++ etc). As linguagens de baixo-nível ou linguagem de máquina são ininteligível pelo ser humano e são executadas diretamente pela máquina (e.g., Assembler).

[7] A tradução do termo *schedules of reinforcement*, como esquemas de reforço, foi sugerida por Azzi, Rocha e Silva, Bori, Fix e Keller (1963) e até hoje é utilizada. Entretanto, não será seguida neste trabalho. O termo esquema, como tradução para *schedule*, frequentemente não é encontrada nos dicionários inglês-português. O termo inglês para "esquema" seria *schema* ou *scheme* (cf. Galvez, 2006). Propõe-se traduzir o termo, então, como programas de reforço, pois parece descrever mais fielmente o seu significado, uma vez que programas

inexistentes, o desenvolvimento de uma nova versão para o programa ProgRefv3 se mostra oportuna. O *software* será uma reformulação do *software* ProgRef v3 (Costa & Banaco, 2002; 2003). Um dos principais motivos dessa reformulação é a portabilidade do novo *software*, ou seja, ele poderá ser instalado e utilizado em um maior número de plataformas (e.g., Windows® XP, Vista e 7). Cuidado especial foi tomado quanto à clareza e lógica do código-fonte. A nova programação possui uma documentação que deverá facilitar sua manutenção (i.e., identificação e correção de possíveis falhas), a implementação de novos recursos no *software*, bem como o desenvolvimento de outros *software*s para o estudo de programas de reforço.

Outra revisão importante realizada no ProgRef v4 foi quanto ao tratamento dos dados coletados, a nova versão possui uma maior flexibilidade de análise por parte do experimentador, além de melhores mecanismos de segurança contra perdas acidentais de dados. Os dados são coletados e podem passar por filtros para análise, tornando possível ao experimentador fazer análises comparativas tanto intra quanto entre sessões com a ajuda de um *software* de análise (ProgRef_DA) desenvolvido como parte do projeto. Além das análises realizadas por este *software*, os pesquisadores também podem gerar tabelas e utilizar um *software* comercial, como o Microsoft Excel®, atendendo assim a sua necessidade específica de apresentação ou formatação das tabelas de resultado. O gerador de gráficos foi reformulado e possibilita uma maior variedade de configurações, facilitando o preparo dos gráficos para gerar figuras para apresentações e publicações. Uma análise do *software* ProgRef v3, seus alcances e limites, será realizada no Capítulo 3 do presente trabalho.

A proposta do presente trabalho, além da defesa e justificação da construção de uma nova versão do *software* ProgRef v3, foi suscitar a discussão do uso de tecnologia em laboratórios de Análise Experimental do Comportamento.

Visando um estudante neófito, o Capítulo 1 define o comportamento segundo um modelo de seleção pelas consequências; apresenta uma descrição dos programas de reforço (como um arranjo experimental “ideal” para o estudo da seleção operante) e os padrões tipicamente gerados; apresenta algumas das críticas que os pesquisadores da área recebem sobre os estudos acerca dos programas de reforço (especialmente a da “simplificação”) e discute essa posição em comparação à de outras ciências; discorre sobre o uso histórico da tecnologia nos laboratórios da Análise Experimental do Comportamento (especialmente no desenvolvimento do registrador cumulativo, salientando que o avanço da

---

de reforço descrevem como eventos antecedentes e consequentes estão programados para ocorrer em função de uma dada resposta do organismo.

tecnologia em outros campos do conhecimento e seu uso pelos analistas do comportamento pode fazer avançar essa área do conhecimento); descreve alguns programas de reforço (especialmente aqueles configuráveis pelo ProgRef) e descreve pesquisas que utilizam de arranjos experimentais com programas de reforço, apontando sua relevância para questões que estão além do laboratório. Esse capítulo pretende responder as seguintes questões: (1) O que são programas de reforço? (2) O estudo de programas de reforço (ainda) é importante? A resposta de ambas as questões parecem fundamentais para justificar a construção de uma nova versão de um *software* (ProgRef) que permite o desenvolvimento de pesquisas em AEC por meio do arranjo de programas de reforço com humanos.

O Capítulo 2 faz um breve histórico do desenvolvimento da informática e sua evolução no intuito de familiarizar o leitor com alguns dos termos utilizados pela tecnologia da informática. Esta familiarização pode ajudar a entender algumas questões relativas às escolhas feitas para a nova versão do ProgRef. É apresentado um breve histórico das linguagens de programação (e.g., BASIC, FORTRAN), discutindo como as mudanças nessa área ocorreram em virtude de problemas específicos de diversas áreas científicas. Com esse histórico e discussão espera-se esclarecer o leitor sobre a escolha de um método programação orientada a objetos (POO) para a programação da nova versão do ProgRef (em vez de uma programação procedural). O capítulo também abordará o uso da informática em outras disciplinas científicas, como a Matemática e a Biologia. Esse enfoque servirá para pontuar como o uso de tecnologias pode auxiliar uma dada área do conhecimento científico, dando suporte a argumentação, que permeia o presente trabalho, de que os investimentos em tecnologia (em geral) e em instrumentos computadorizados para coleta de dados (em particular) podem fazer avançar o ensino e a pesquisa em Psicologia e Análise do Comportamento. São descritos alguns *software*s desenvolvidos para pesquisa e aplicações em Psicologia (Etholog 2.2; Mestre; DRL; Med-PC) procurando levantar alguns aspectos aparentemente positivos ou negativos de cada um com vistas a sugerir direções para a programação da nova versão do *software* ProgRef. Algumas questões relativas à escolha de uma linguagem de programação são discutidas. Faz-se uma diferenciação entre as linguagens de programação para propósitos gerais (e.g., Visual Basic; Delphi; C++) e as linguagens de programação para propósitos específicos (como é o caso do Med-PC) discutindo-se as vantagens e desvantagens de cada uma delas. Essa discussão sobre as linguagens de programação procuram justificar a escolha do Visual Basic®.NET para a programação do ProgRef v4. As perguntas que o capítulo pretende responder são: (1) O avanço da tecnologia (especificamente da informática) teve impacto na Psicologia? (2) Quais sistemas

computadorizados podem ser citados como exemplo de instrumentos de pesquisa ou aplicação na Psicologia? E quais são os aspectos positivos e negativos de cada um deles? (3) Que fatores podem (ou devem) ser levados em conta na escolha de uma linguagem e de um método de programação? (4) Com relação ao método de programação, o que é e quais as vantagens de uma programação orientada a objeto (POO)? As respostas às questões (1) e (2), entre outras coisas, buscam justificar o investimento no aprendizado de uma linguagem de programação, especialmente na formação de pesquisadores em AEC. As respostas às questões (3) e (4) buscam justificar algumas das escolhas feitas para o desenvolvimento da nova versão do ProgRef.

O Capítulo 3 apresenta uma análise crítica do *software* ProgRef v3 (Costa & Banaco, 2002; 2003). Essa análise crítica pretende levantar alguns pontos sensíveis no projeto do ProgRef v3 e, assim, evitá-los no desenvolvimento do ProgRef v4. O capítulo descreve de maneira resumida alguns recursos do ProgRef e, ao mesmo tempo, descreve pesquisas (com o ProgRef ou não) que justificam a inclusão (ou manutenção) de um determinado recurso no *software*. A descrição de algumas pesquisas com a utilização do *software* ProgRef v3 também indicam a relevância do instrumento em gerar pesquisas, fornecendo justificativas adicionais para a construção da nova versão do *software* que incorporou os recursos existentes e criou novas possibilidades tanto no uso (e.g., utilização com versões mais atuais do Windows, além das antigas) quanto na configuração de novos arranjos experimentais (e.g., a inclusão de programas de reforço tandem e DRH). As questões que permeiam este capítulo são: (1) O que é e o que faz o ProgRef v3? (2) De maneira geral, quais recursos estão disponíveis e o que justifica a manutenção desses recursos? (3) A construção do ProgRef v3 propiciou a realização de pesquisas que justifiquem que uma nova versão tenha sido pensada e construída? A questão (2), juntamente com as duas últimas questões levantadas em relação ao capítulo anterior, esclarecem alguns pontos que influenciaram decisões mais específicas tomadas ao longo do desenvolvimento do ProgRef v4.

O Capítulo 4 aborda a construção do *software* ProgRef v4 propriamente dito. O capítulo descreve, resumidamente, a “evolução” do Windows® e a necessidade de atualização de *software*s (o que inclui o ProgRef). O capítulo procura apresentar o projeto do ProgRef v4 à luz dos problemas encontrados no ProgRef v3 e dos recursos de programação disponíveis. É retomada a discussão acerca da escolha da POO e sua "flexibilidade". Descrevem-se alguns recursos disponíveis no Visual Studio® (pacote de programação que inclui o Visual Basic.NET) que podem facilitar a construção da nova versão, bem como pavimentar o caminho para a inclusão, futura, de novos recursos ao *software*. O capítulo

também apresenta a descrição de utilização do *software* de pesquisa ProgRef v4 e do ProgRef_DA v1. Essa parte do capitulo é de cunho estritamente didática para apresentação dos recursos disponíveis nos programas. Um pesquisador poderá utilizar como um manual introdutório.

# CAPÍTULO 1

## PROGRAMAS DE REFORÇO

A Análise do Comportamento define seu objeto de estudo – o comportamento – como a interação organismo-ambiente e o modelo de causalidade adotado é o modelo de seleção pelas consequências (Skinner, 1984; Chiesa, 1994). Os programas de reforços constituem excelentes arranjos para se estudar o comportamento assim definido.

Para o leitor não familiarizado com a área, o presente capítulo definirá o que são programas de reforço e descreverá alguns deles, que poderão ser utilizados no *software* ProgRef v4. Com um cunho histórico, o capítulo analisará e descreverá o uso da tecnologia na Análise Experimental do Comportamento (AEC) fazendo um paralelo entre avanços da tecnologia e a sua utilização pela AEC.

O comportamento de organismos em seu ambiente natural é extremamente complexo e sofre influência de diversas variáveis. Para estudar como as consequências relacionam-se com um determinado comportamento o pesquisador precisa de uma tecnologia que forneça grande adaptabilidade para diversos problemas de pesquisa. Segundo Ator (1991), Skinner desenvolveu instrumentos básicos para o estudo sistemático das relações comportamentais do organismo com o seu ambiente. O ambiente experimental que ele construiu para as pesquisas em laboratório foi denominado caixa de condicionamento operante (também conhecida como Caixa de Skinner) e permite simular o processo de seleção por consequências dentro de um laboratório utilizando diversos programas de reforço.

Um “programa de reforço” é uma descrição completa das condições ambientais e comportamentais que estão presentes quando uma resposta é seguida de uma consequência (Donahoe & Palmer, 1994; Ferster & Skinner, 1957; Morse, 1966). Ou seja, programas de reforço podem ser definidos como regras que especificam as relações entre respostas e suas consequências. As consequências podem ser programadas para ocorrerem após (1) predeterminado número de respostas, conhecidos por programas de razão; (2) predeterminado intervalo de tempo tiver transcorrido, conhecidos por programas de intervalo; e (3) combinação de intervalo de tempo e respostas tiver ocorrido, como por exemplo, programas que dependem da taxa ou do espaçamento temporal das respostas prévias, programas que reforçam diferencialmente a taxa ou tempos entre as respostas (Catania, 1998; Ferster & Skinner, 1957; Lattal, 1991).

Os programas de reforço básicos são descritos por diversos autores, facilmente encontrados em livros ou capítulos de livros (introdutórios ou não) para estudo da Análise Experimental do Comportamento (e.g., Catania, 1998; Ferster & Skinner, 1957; Ferster, Culbertson & Boren, 1979; Lattal, 1991; Millenson, 1967/1975; Souza Júnior & Cirino, 2004). Os programas de reforços básicos são, geralmente, divididos em intermitentes e contínuos. Apenas dois programas não-intermitentes são geralmente descritos. Um deles é o reforço contínuo (CRF)[8], no qual todas as respostas emitidas são seguidas por uma dada consequência e o outro é a extinção (EXT), no qual nenhuma resposta emitida é seguida por uma dada consequência.

Nos programas de reforço intermitentes a consequência não é programada para seguir a emissão de cada resposta, mas apenas a emissão de algumas respostas[9]. Em um programa de razão fixa (FR) a resposta que completa uma razão predeterminada é seguida por uma dada consequência. Por exemplo, em um programa em FR 5, a quinta, a décima, a décima quinta resposta (ou seja, as respostas múltiplas de 5) serão seguidas por uma dada consequência programada. Quando o número de respostas, requerido para que o reforço[10] seja apresentado, variar de um reforço para outro, o programa é denominado de Razão Variável (VR). O reforço é liberado após um dado número médio de respostas que estão entre dois valores extremos arbitrários, os valores normalmente são obtidos de forma randômica previamente ao experimento. Por exemplo, a consequência poderia ser programada para ocorrer na 2ª; 8ª; 14ª; 23ª; 37ª e 66ª resposta (sempre contada a partir da liberação do último reforçador ou do início da sessão experimental). A média aritmética dos seis valores (2 + 8 + 14 + 23 + 37 + 66 = 150; 150/6 = 25) indica o valor da VR. Neste exemplo, temos um VR 25, que nos indica que será liberado um reforçador para cada 25 respostas em média. Geralmente os valores não são arranjados em ordem crescente, mas randômica.

---

[8] Após o termo de um programa de reforço, será apresentada entre parênteses a sigla que representa o programa. A sigla é formada tendo como base o termo em inglês. Esse estilo é amplamente utilizado na bibliografia da Análise Experimental do Comportamento. Para uma discussão e uma exemplificação mais exaustiva dos programas de reforço ver, por exemplo, Catania (1998, pp. 191-192).

[9] Neste contexto, o termo "resposta" refere-se à resposta-alvo que está sendo estudada (e.g., pressão à barra). O organismo emitirá diversas respostas durante um experimento, mas apenas a resposta-alvo é considerada para o cumprimento das exigências de um programa de reforço.

[10] Um evento é chamado "reforçador" quando ele é produzido por uma dada resposta e quando ele aumenta a frequência de emissão de respostas dessa classe no futuro. Portanto, "reforço" é a descrição de uma relação. Por exemplo, a liberação de uma gota de água pode tornar-se reforçadora em certas condições e, neste caso, podemos chamar a gota de água de "reforçador". Isso não significa que uma gota de água seja um evento reforçador em qualquer circunstância. Não há nenhuma propriedade física que defina um reforçador (ver Catania, 1998, Capítulo 5). Entretanto, em alguns pontos do presente trabalho, os termos "reforço", "reforçador" ou algum termo derivado desses, serão utilizados para se referir à(s) consequência(s) produzida(s) por uma resposta (embora seus efeitos sejam supostos e não observados).

Os programas de intervalo podem ser de intervalo fixo (FI) ou intervalo variável (VI). A probabilidade de a resposta ser reforçada é alterada pela simples passagem do tempo. No FI a primeira resposta emitida após um período designado de tempo é seguida por uma dada consequência. Por exemplo, em um FI 10 s, a consequência é liberada para a primeira resposta que for emitida após a passagem de 10 segundos (desde o último reforçador liberado ou do início de uma sessão experimental). Respostas que ocorrerem antes de terminado este intervalo não têm consequências programadas. No VI, o período de tempo para que a resposta tenha uma dada consequência varia de acordo com uma série de valores, no qual o valor médio é apresentado para definir o programa em vigor. Os valores apresentados anteriormente no exemplo do VR poderiam ser utilizados para se construir um programa de VI. Para isso, basta que os valores representem o intervalo de tempo em vez do número de respostas. Portanto, utilizando os valores do exemplo do VR poderíamos arranjar um VI 25 s.

Outros arranjos ambientais podem ser programados, por exemplo, o programa de reforço diferencial de baixa taxa [*differential reinforcement of low rates*] (DRL), no qual apenas respostas emitidas com intervalo entre respostas (IRT) maiores que um determinado valor são seguidas por uma determinada consequência. Por exemplo, em um DRL 10 s, a consequência é liberada para a primeira resposta que for emitida após a passagem de 10 segundos (desde a última resposta emitida). Entretanto, se alguma resposta for emitida antes de terminado este intervalo, o cronometro é zerado. Ou seja, cada resposta com IRT < 10 s "adia" o reforço. O programa de reforço diferencial de alta taxa [*differential reinforcement of high rates*] (DRH) é aquele no qual, apenas respostas emitidas com intervalo entre respostas (IRT) menores que um determinado valor são seguidas por uma determinada consequência. Por exemplo, um DRH 0,5 s poderia ser arranjado especificando-se que a consequência fosse liberada se as 10 últimas respostas ocorrerem em, no máximo, cinco segundos (para que isso ocorra é preciso que as 10 respostas sejam emitidas com um intervalo médio de 0,5 s).

Dois outros programas que envolvem intervalo de tempo, mas que, diferentemente dos programas descritos acima, não envolvem relação de contingência[11] com a resposta do organismo é o programa de tempo fixo (FT) e tempo variável (VT). Em um programa FT, após a passagem de um determinado período de tempo, o "reforço" é liberado independente da emissão de qualquer resposta-alvo. Por exemplo, em um FT 30 s, um evento é programado para ocorrer (e.g., liberação de uma gota de água, de uma pelota de comida ou

[11] O termo contingência refere-se às relações de controle (ou de dependência) entre eventos. De acordo com Catania (1998, p. 74) as contingências são "o efeito de uma resposta sobre a probabilidade de um estímulo".

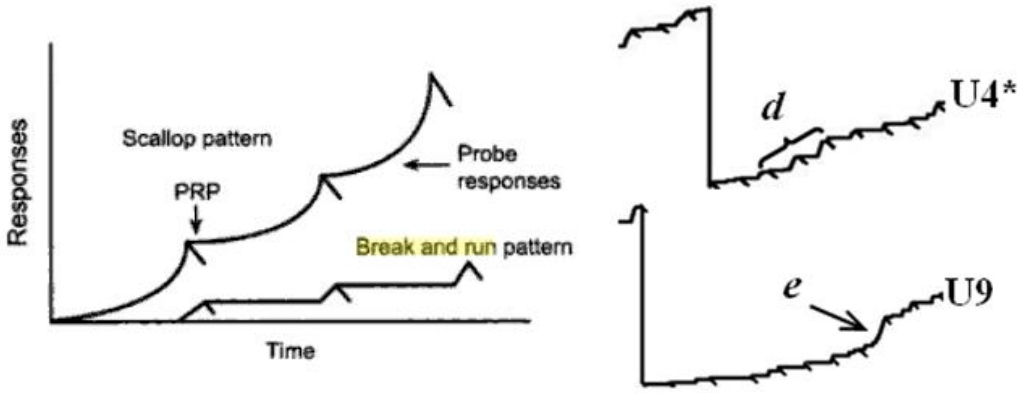
de pontos em um contador) a cada 30 segundos. Em um programa VT o período de tempo para a ocorrência desse evento é variável de acordo com uma série de valores pré-definidos, o valor médio dessa série define o programa. Essa série pode ser calculada da mesma forma que no VI, com a diferença que no VI exige-se a emissão de uma dada resposta para a ocorrência do evento (reforçador) e no VT a resposta não é exigida.

Nove dos 10 programas de reforço descritos aqui podem ser arranjados no *software* ProgRef v3 (CRF; Extinção; FR; VR; FI; VI; FT; VT e DRL) e os 10 programas (aqueles citados mais o DRH) estão disponíveis na nova versão do *software* ProgRef. Uma característica dos programas de reforço – perceptível das descrições fornecidas acima – é sua lógica, que os torna altamente convenientes à programação computadorizada. Millenson (1967/1975) e Lattal (1991), por exemplo, exibem representações diagramáticas desses programas de reforço que facilitam, não apenas o entendimento da configuração de um programa de reforço, mas também o desenvolvimento de uma lógica para a programação computadorizada desses programas de reforços.

Outra característica, ainda mais importante, é que cada programa de reforço produz um padrão e uma taxa de respostas particular. As características qualitativas e, até mesmo, algumas propriedades quantitativas do desempenho controlado pelo programa são iguais para muitas diferentes espécies, respostas e reforços. Isso possibilita a predição de como um determinado organismo irá se comportar ou a identificação de a qual programa de reforço o organismo está submetido (Chance, 2009; Nevin, 1979).

A Figura 1.1 apresenta padrões comportamentais verificados com sujeitos submetidos a um programa de FI. A figura da esquerda é uma curva idealizada. O padrão de *scalloping* é caracterizado por uma pequena pausa pós-reforço [*pause after reforcement*] (PRP) seguido de um responder positivamente acelerado, verificado pelo formato de “concha” na curva. O padrão de *break and run* é caracterizado por uma longa pausa pós-reforço (PRP) seguido de um “jorro” [*burst*] de respostas (respostas com um pequeno IRT) no final do intervalo do FI. Esse padrão é verificado, normalmente, após uma longa exposição em FI (Pierce & Cheney, 2004). À direita da Figura 1.1 são exibidos seis minutos dos registros cumulativos de dois universitários em FI 30 s. A letra *d* indica um padrão de *break and run* e a letra *e* um padrão de *scallop*. Esses registros mostram que esses padrões comportamentais são encontrados em humanos e que, assim como com organismos não-humanos, esses padrões não se repetem em cada intervalo entre reforços (IRI). Os registros cumulativos exibidos à direita na Figura 1.1 foram geradas pelo *software* ProgRef v3.

**Figura 1.1** - Padrões de respostas de organismos submetidos a FI. À esquerda é apresentado um gráfico, extraído de Pierce e Cheney (2004, p.133, Figura 5.9) com dados hipotéticos de dois padrões (*scallop* e *break and run*). À direita são exibidos seis minutos dos registros cumulativos de dois universitários em FI 30 s. A letra *d* indica um padrão de *break and run* e a letra *e* um padrão de *scallop*




**Fonte**: dados extraídos de De Freitas (2009, p. 28, Figura 7).

A importância dos padrões obtidos pelos programas de reforço é reconhecida por Sidman (1960) que sugere o uso de programas como um teste de adequação de um laboratório. Caso os padrões, amplamente apresentados pela literatura, não fosse obtido pelo experimentador alguma variável não conhecida estaria interferindo no controle experimental. Sidman também aponta para o uso de programas de reforço como linha de base para investigar os efeitos comportamentais de outras variáveis.

O comportamento dos organismos é afetado pela experiência prévia ou o comportamento pode ser explicado recorrendo-se "apenas" às variáveis presentes no momento que o organismo se comporta? Essa é uma pergunta bastante geral e sua resposta têm desdobramentos bastante complexos. No momento, no interesse imediato da presente discussão, o ponto importante é saber se utilizando arranjos experimentais com programas de reforço podemos, pelo menos, começar a responder essa questão. O estudo de Weiner (1964) sugere que sim e que podemos começar a fazer isso utilizando alguns dos programas de reforço simples descritos até aqui.

Weiner (1964) investigou experimentalmente o efeito de duas diferentes histórias sobre o comportamento subsequente em FI. Seis participantes foram distribuídos em dois grupos. Durante a Fase 1 (Fase de construção da história), os participantes do Grupo 1 foram submetidos à contingência de FR, enquanto os participantes do Grupo 2 foram expostos à contingência de DRL. Os participantes do Grupo 1 emitiram altas taxas de respostas em FR e os participantes do Grupo 2 emitiram baixas taxas de respostas em DRL. Posteriormente, na

Fase 2 (Fase de teste), os participantes de ambos os grupos foram expostos à contingência de FI. Como resultado geral dessa fase, os participantes do Grupo 1 emitiram taxas de respostas mais altas do que os participantes do Grupo 2. Estes resultados sugerem que humanos expostos a diferentes histórias de reforço comportam-se diferentemente em FI. Resultados semelhantes foram obtidos por Urbain, Poling, Millam e Thompson (1978) com ratos.

Portanto, respondendo parcialmente a questão inicial, o comportamento dos organismos pode ser afetado pelas experiências prévias. As condições sob as quais isso é mais ou menos provável de ocorrer dependem, é claro, do desenvolvimento de uma ampla linha de pesquisa sobre o assunto que ultrapassarão, inclusive, o uso de programas de reforço como linha de base. Com base apenas na pesquisa descrita pode-se dizer apenas que a frequência com que um comportamento ocorre *pode* ser afetada pela sua história. Claro, que outras variáveis também podem afetar a frequência de um comportamento e, claro, no estudo descrito foi utilizada uma resposta simples em um contexto relativamente simplificado.

Segundo Chance (2009), uma das críticas ao uso de programas de reforço para o estudo do comportamento é que eles não passam de construções artificiais, não encontradas no mundo real. O autor contrapõe esta crítica argumentando que o objetivo é descobrir regras que descrevem a forma que o ambiente interage com o organismo e afeta o comportamento. Seria muito difícil o estudo do comportamento sem a simplificação nas variáveis. Essa dificuldade não é característica da Análise Experimental do Comportamento podendo ser vista em outras ciências como na Física ou Economia. Por exemplo, em laboratórios de física é comum o uso de trilhos de ar para se estudar as leis de Newton. Este trilho de ar permite que se realize uma série de experimentos nos quais a força de atrito é reduzida, sendo possível obter o comportamento de um objeto móvel mais próximo da idealização teórica, facilitando o ensino e o entendimento de um fenômeno (Pimentel, Zumpano & Yaginuma, 1989). Este é um caso em que é utilizada uma simplificação de variáveis com o intuito de diminuir os efeitos que o atrito teria sobre o deslocamento de um corpo para que o estudo seja realizado e previsões sejam propostas para experimentos em mecânica (Laudares, Lopes & Cruz, 2004).

Outro exemplo de simplificação pode ser encontrado na Economia. Parte da Economia está preocupada com a previsão do comportamento de um “homem médio”. Por exemplo, em uma praia com sol forte é previsível que aumente o consumo de cerveja. Essa previsão geral não esta preocupada com o fato de que o indivíduo A ou B não irão consumir cerveja por motivos religiosos. Outra simplificação utilizada pelos economistas é a idealização do homem econômico, que por definição sempre se comporta de forma a

maximizar seus ganhos e reduzir suas perdas. A microeconomia utiliza-se da condição *coeteris paribus*, ou seja, hipótese que pressupõe que todas as demais condições que possam influenciar no relacionamento entre duas variáveis, funcionalmente dependentes, sejam mantidas constantes (Garófalo, 2004).

Desse modo, tanto a Economia quanto a Física, entre outras ciências, utilizam procedimentos "simplificados" que permitem a investigação de seu objeto de estudo, fornecendo dados para análise e uma melhor compreensão da teoria proposta, mesmo quando as condições de interesse são complexas. Para realizar investigações na área da Análise do Comportamento, os programas de reforço possuem uma grande variedade de arranjos, sendo possível especificar os comportamentos e outros eventos ambientais que são igualmente diversificados e flexíveis. É essa flexibilidade – característica dos programas de reforço (cf. Zeiler, 1984) – que permite que os programas de reforço possam ser utilizados como modelos comportamentais em diversos campos de interesse como na farmacologia, economia, biologia comportamental etc.

Conforme sugeriu Lattal (1991), a maioria dos experimentos comportamentais requer uma situação relativamente simples, na qual um estímulo seguramente controla a taxa de repostas ou padrões comportamentais. Os diferentes arranjos possíveis utilizando programas de reforços, assim como formas para registrar os desempenhos nesses programas, dependem apenas do problema de pesquisa e da experiência do pesquisador.

Em resumo, programas de reforço descrevem relações entre respostas e eventos ambientais (antecedentes e consequentes); estas relações podem ser arranjadas em ambientes de laboratório utilizando respostas relativamente simples (como pressionar uma barra, pressionar um botão, bicar um disco, puxar uma corrente etc.) em contextos relativamente simples, mas altamente controlados. Esse controle visa reduzir o efeito de variáveis estranhas ao problema experimental em investigação, para que a mudança comportamental observada possa ser atribuída à variável experimental manipulada. A escolha por respostas relativamente simples (e de baixo custo para o organismo) favorece a sua reprodutibilidade e, portanto, a relação entre eventos antecedentes, a resposta e as consequências podem ser arranjadas para ocorrer e pode ser observadas centenas ou milhares de vezes em um período de tempo relativamente curto.

Uma forma de registro do comportamento que foi amplamente utilizado nos estudos envolvendo arranjos experimentais com programas de reforço foi o registro cumulativo, que exibe a taxa de respostas momento a momento em uma sessão experimental.

A taxa de respostas tornou-se um dado importante justamente porque Skinner estava interessado no processo de mudança do comportamento e apoiou-se em manipulações experimentais para teste de estabilidade da unidade de resposta. A taxa de respostas como um dado experimental supria – e ainda supre – muito bem esse objetivo. Esse dado era facilmente coletado através de registradores cumulativos (Skinner, 1966).

Os registradores cumulativos possibilitavam o monitoramento da sessão em tempo real, ou seja, através das marcas registradas era possível verificar as alterações comportamentais, produto da contingência em vigor, momento a momento (os gráficos exibidos na Figura 1.1 são exemplos de registros cumulativos). Lattal (2004) afirmou que, em um arranjo experimental comum, o registrador cumulativo era capaz de executar quatro funções: (1) cada resposta movimentava verticalmente (*step*) a ponteira de registro através do papel; (2) se assim programado, a ponteira de registro marcava com uma breve deflecção, normalmente para baixo, quando ocorria um reforço; (3) quando a ponteira de registro atingisse o topo da página, ela retornava para a parte de baixo, função conhecida como *reset* da ponteira; e (4) quando ocorresse algum evento ambiental significativo, tais como respostas selecionadas e/ou alteração estímulos, poderia ser programado uma segunda ponteira de registro para defletir para cima ou para baixo indicando assim o momento em que o evento ambiental ocorreu.

Ao utilizar o registrador cumulativo, tendo disponíveis essas quatro funções, o investigador tem acesso visual às alterações na frequência do comportamento em tempo real (como nenhuma outra medida tentada anteriormente), revelando efeitos comportamentais moleculares, ou seja, sutis e localizados. Além disso, os registros cumulativos revelam propriedades mais molares do comportamento, como a característica temporal e padrões gerados em cada programa de reforço (Catania, 1998; Lattal, 2004).

No entanto, segundo Lattal (1991), o uso dos registros cumulativos está sendo menos frequente nos artigos científicos da Análise Experimental do Comportamento, substituídos por dados mais quantitativos (índices de desempenho). Essa utilização de dados mais quantitativos tem grande influência do paradigma molar de análise. Alguns índices utilizados são: a taxa de respostas, a taxa de "*running*", intervalo entre reforços (IRI), pausa pós-reforço (PRP). Essas medidas descrevem efeitos comportamentais globais – ou molares – das variáveis independentes. A taxa de respostas cumulativa, a taxa média de resposta e a PRP podem ser enriquecidas com medidas mais locais ou moleculares que descrevem padrões e desempenhos dentro de porções da sessão. A distribuição do tempo entre as respostas (IRT), isto é, entre respostas sucessivas é uma maneira útil de descrever padrões de respostas.

Esses modos de análise, o olhar molar e o molecular, diferem de forma paradigmática e não teórica. Não são os dados, nem o experimento, quem decide entre um ou outro. A interpretação do mesmo fenômeno (i.e., dos mesmos dados) difere, a depender da adoção de uma análise molar ou molecular. Esses diferentes modos de interpretação são incomensuráveis no sentido *khuniano* (Baum, 2002). Historicamente a Psicologia do século XIX apresentava um olhar molecular (olhar atomista) para os fenômenos em estudo, pois era baseada em explicações do comportamento que poderiam ser formuladas pelo pensamento de pequenas unidades discretas, sendo em seguida reunida em unidades maiores – como a união de átomos dentro das moléculas na química. A visão molecular conta com eventos e causas momentâneos, que levam a postular eventos e causas hipotéticas quando não são aparentes. Os pontos principais na análise molecular é a utilização de eventos discretos, momentâneos e contíguos entre si. A visão molar é baseada nos conceitos de agregação e amplos padrões do comportamento. Padrões estendidos temporalmente. Assim, instantaneamente o comportamento não pode ser medido no momento que supostamente ocorreu, mas pode ser inferido de um padrão mais amplo (Baum, 2002, 2003).

Os programas de reforço, que possibilitavam o arranjo experimental de diversas contingências, os instrumentos para controle de variáveis (Caixa de Skinner) e de registro de eventos comportamentais (registrador cumulativo) foram inicialmente utilizados para desenvolver pesquisas utilizando organismos não-humanos (e.g., pombos, ratos etc.). Com o decorrer dos anos pesquisas utilizando participantes humanos foram realizadas com os mais variados propósitos (Lattal, 2006). Segundo Shull e Lawrence (1998), é possível apontar quatro objetivos principais nas pesquisas envolvendo programas de reforço com humanos: (1) estabelecimento de linha de base comportamental com uma propriedade particular, possibilitando a comparação e visualização dos efeitos de outras variáveis; (2) comparação entre humanos e não-humanos dos padrões comportamentais apresentados em cada programa de reforço; (3) delineamentos experimentais com humanos para testar conceitos teóricos e/ou revelar relações funcionais gerais; e (4) utilização dos programas de reforço para estudar comportamentos humanos complexos (e.g., resolução de problemas, aprendizagem por instrução, ajustes comportamentais ótimos; percepção da causalidade entre outros).

Essas pesquisas têm usado paradigmas experimentais de comportamentos complexos demonstrando que esses modelos possibilitam a investigação sistemática de variáveis que afetam esses comportamentos. Catania (1998) defendeu que programas de reforço podem ser utilizados para explorar propriedades de comportamentos complexos. Neste contexto, ele apresenta o conceito de síntese comportamental. “Sintetizar” um

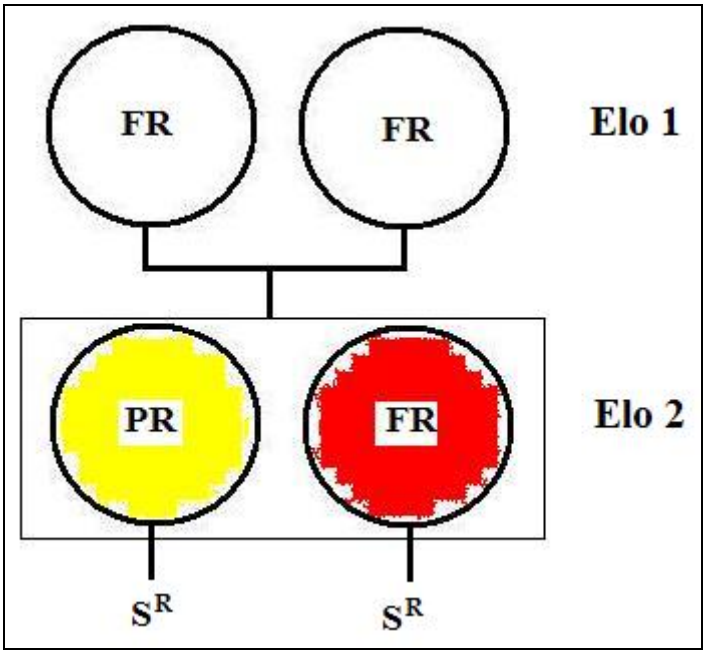
comportamento seria “construir” um comportamento complexo no laboratório que mantivesse as relações funcionais encontradas nesse comportamento quando observado no ambiente fora do laboratório.

A “síntese comportamental” seria feita arranjando programas de reforços simples em configurações mais complexas. Um exemplo de síntese comportamental é o estudo de Wanchisen, Tatham e Hineline (1988), no qual o estudo do comportamento de forrageio de pássaros foi estudado em laboratório, utilizando-se de um arranjo experimental com programas de reforço. O comportamento de forrageio pode ser definido como o deslocamento dos animais de uma fonte de alimento para outra, no ambiente selvagem, permanecendo em uma área ou mudando para outra, dependendo dos recursos alimentares disponíveis nesse local.

Para atingir os propósitos do estudo foram utilizados programas concorrentes encadeados. Nos programas concorrentes, dois ou mais programas simples são correlacionados com *operanda* separados, que estão disponíveis simultaneamente. Nos programas encadeados, dois ou mais componentes de programas simples são correlacionados com estímulos distintos. No programa concorrente encadeado, cada componente é nomeado como elo [*link*] e os elos (programas) operam sucessivamente e ocorre o reforço apenas no elo final, ou seja, após os requisitos de cada componente terem sido atingidos. No programa concorrente encadeado, no elo inicial são apresentados dois ou mais programas, e o cumprimento do requisito do programa escolhido têm como consequência a apresentação do elo seguinte sucessivamente (Lattal, 1991).

A Figura 1.2 ilustra o arranjo de um programa concorrente encadeado. No primeiro elo dois programas FR estão programados concorrentemente. A cor dos dois discos, neste exemplo, é branca. Quando a razão é cumprida, em algum dos dois discos, a consequência é a mudança para o segundo elo da cadeia, em que as cores dos discos passam a ser amarela e vermelha nos quais operam um programa PR e um FR, respectivamente. Quando um dos dois programas é cumprido o reforçador é liberado.

**Figura 1.2** - Diagrama do arranjo de um programa de reforço concorrente encadeado.




A analogia referente ao fenômeno de forrageio, utilizando os programas concorrentes encadeados no estudo de Wanchisen et al. (1998) é: (1) variar os programas de reforço nos elos iniciais é análogo a variar o tempo e o esforço envolvidos em viajar de uma fonte de alimento para outra; (2) variar os programas nos elos terminais é análogo a variar a disponibilidade ou depredação de fontes diferentes de alimento em locais diferentes. Pombos foram expostos a um programa de reforço concorrente encadeado. Duas chaves ficavam inicialmente brancas e o programa em vigor era um FR 1. Depois de emitida uma resposta, as chaves eram iluminadas. Uma com a cor vermelha, na qual o programa em vigor era o de FR e uma na cor amarela, na qual o programa em vigor era o de razão progressiva PR 20, ou seja, a cada reforço a razão (número de respostas necessárias para liberação do reforço) era acrescida de 20 respostas. Os valores do FR (no Elo 2) variavam a cada cinco sessões (em média) com valores crescentes e decrescentes. Os principais resultados obtidos foram: (1) houve uma tendência em escolher o FR quando valores do PR excederam o ponto de equivalência entre FR e PR; (2) essa escolha foi mais frequente e extrema em valores de PR altos do que em baixos, ou seja, a probabilidade de escolher o componente em que o programa de FR estava em vigor aumentava à proporção que o valor do PR aumentava.

Para se aproximar mais do ambiente “natural”, Wanchisen et al. (1988) propuseram um segundo experimento. A diferença do experimento anterior era a capacidade de simular a recuperação em termos de alimentação do ambiente depredado, ou seja, seguindo

a analogia o valor do programa PR era reiniciado após uma escolha no programa FR. O procedimento foi realizado como descrito no experimento anterior, com a seguinte diferença: a cada programa de FR completado (no Elo 2) o valor de PR voltava para zero. O principal resultado obtido foi que o modelo de mudança deste experimento sugere claramente uma sensibilidade para as condições de reinício do PR. Em curto prazo entre a escolha de um PR de razão baixa (analogia de esgotamento de recursos) e de um FR de razão alta (analogia de viagem para um novo *habitat*), sujeitos frequentemente escolhem o FR. Os resultados desse estudo sugerem que arranjos formais (arranjados pela configuração de programas de reforços complexos) podem simular situações de esgotamento de recursos fora do laboratório (como aqueles observados no comportamento de forrageio em pombos ou mesmo outras situações de esgotamento de recursos).

Outro exemplo de síntese comportamental é o estudo apresentado por Rachlin e Green (1972). Eles estavam interessados no estudo do "comprometimento" como um comportamento de escolha e também em verificar se um modelo matemático seria capaz de prever a alteração da preferência (de uma situação de baixa densidade de reforço imediato para uma de alta densidade de reforço, mas com atraso). Para sintetizar esse arranjo em um laboratório, os autores utilizaram programas concorrentes encadeados. Cinco pombos foram expostos a um programa encadeado. A caixa experimental tinha duas chaves que podiam ser iluminadas. No elo inicial elas permaneciam brancas e um programa de FR 25 estava em vigor simultaneamente: a vigésima quinta resposta emitida, na presença das duas chaves, determinava a escolha. As chaves trocavam de cor após a passagem de um tempo (T) dependendo da resposta de escolha. Caso fosse escolhida a chave da esquerda, após T, as chaves alteravam de cor para verde e vermelha (escolha "livre"). Ao responder na chave verde, o sujeito tinha como consequência acesso ao comedouro por dois segundos (baixa densidade de reforço). Na chave vermelha o acesso ao comedouro era de quatro segundos (alta densidade de reforço), ou seja, a densidade de reforço dobrava, mas após um atraso de quatro segundos. Caso o pombo no elo inicial emitisse a resposta de escolha na chave da direita, após T, uma das chaves era desligada e a outra ficava verde (escolha "forçada"). Ao responder nessa chave o sujeito tinha acesso à alta densidade de reforço atrasado (após quatro segundos, acesso ao comedouro por quatro segundos). A metodologia do estudo foi variar o tempo T para verificar a alteração de preferência entre uma escolha de baixa densidade de reforço imediata, ou uma alta densidade de reforço com atraso e a alteração de preferência entre uma situação que levava a uma escolha "livre" ou "forçada".

Os principais resultados apresentados foram: (1) os sujeitos preferiram uma menor quantidade de reforço desde que fosse imediata; (2) a corroboração do modelo matemático que previa que, conforme T aumentasse, ocorreria uma mudança na preferência. Para T de curta duração, a escolha seria para uma baixa densidade de reforço imediata. Para T de maior duração, a escolha de uma alta densidade de reforço, mesmo com um atraso, era preferível. Os experimentadores sintetizaram um tipo de autocontrole, um comprometimento por um curso de ação com o uso de programas de reforço.

O uso da tecnologia tem favorecido a AEC facilitando e, muitas vezes, permitindo a observação do comportamento e controle de contingências para o seu estudo. O uso do registrador cumulativo é bem representativo quanto a isso. O avanço da tecnologia propicia a utilização de novos instrumentos e a investigação de variáveis que anteriormente não podiam ser investigadas por falta de um adequado controle experimental. Os estudos de laboratórios inicialmente executados em equipamentos mecânicos evoluíram para registros mais precisos (e.g., uso da eletrônica digital), mas os programas de reforço continuam a ser importantes. Alguns exigem uma maior habilidade e domínio de tecnologia para o controle experimental por parte do experimentador. Esse empenho do experimentador pelo conhecimento de novas ferramentas para realizar os estudos do comportamento em laboratório são justificáveis, pois como visto esses estudos podem acrescentar informações para compreensão de padrões comportamentais encontrados em um ambiente natural, como o forrageio e comprometimento.

No próximo capítulo será abordado como a Psicologia tem utilizado a informática para melhorar seu controle experimental e até mesmo a sua intervenção em contextos aplicados.

# CAPÍTULO 2

## TECNOLOGIA, INFORMÁTICA E PSICOLOGIA

A tecnologia da informática teve uma grande evolução em um curto período de tempo, aproximadamente cinco décadas. Delyra (1997) defendeu que se trata de uma revolução da informática, caracterizada por um processo de mudanças quantitativas grandes e rápidas no modo de vida dos indivíduos e qualitativamente diferenciável do modo anterior, ocorrendo uma mudança de paradigma. Pode-se notar um crescimento exponencial das tecnologias da informática, por exemplo, o crescimento da rede mundial de computadores (*Internet*), o número de servidores no início da década de 90 estaria próximo a $10^2$ e no final daquela década estaria próximo de$10^6$.

Delyra (1997) apontou outra singularidade da revolução da informática: a transferência das informações e tecnologias, da comunidade especializada (o grupo que desenvolve as novas tecnologias) para a população em geral, se faz de uma forma muito mais rápida do que em outros campos tecnológicos. O cidadão comum atualmente tem disponível, de uma forma relativamente fácil, várias formas de manipular a informação. Essa argumentação pode ser confrontada com os dados de uma pesquisa sobre acesso à Internet e posse de telefone móvel celular para uso pessoal realizada pelo IBGE em 2005. Quanto ao acesso a Internet, 21% (32,1 milhões) da população (de 10 anos ou mais de idade) acessaram pelo menos uma vez, no intervalo de três meses, a Internet em algum local – domicílio, local de trabalho, estabelecimento de ensino, centro público de acesso gratuito ou pago, domicílio de outras pessoas ou qualquer outro local – por meio de microcomputador. A metade dos internautas utilizou a rede no domicílio em que morava e 39,7% em seu local de trabalho. Os usuários de Internet possuem rendimento médio mensal domiciliar per capita de R$ 1.000,00. A conexão discada se mostrou mais frequente do que a de banda larga (Instituto Brasileiro Geografia e Estatística – IBGE, 2005).

Talvez para a grande maioria da população de um país em desenvolvimento, como o Brasil, o acesso a informação por meio da Internet se mostra difícil. No entanto, é inegável a importância da utilização da informática pelos integrantes de uma determinada área científica. Para familiarizar o leitor com alguns dos termos utilizados pela tecnologia da informática, este capítulo fará um breve histórico do desenvolvimento da informática e sua evolução (ou revolução digital). Um breve histórico das linguagens de programação possibilitará o entendimento de como as mudanças nessa disciplina ocorreram muitas vezes

em resposta a problemas específicos das mais diversas disciplinas científicas e esclarecerá o leitor sobre a escolha de uma linguagem orientada a objetos para a programação do ProgRef. O capítulo também abordará o uso da informática em outras áreas científicas, como a Matemática e a Biologia. Esse enfoque servirá para pontuar como o uso de tecnologias pode auxiliar uma dada área do conhecimento científico, dando suporte a argumentação, que permeia o presente trabalho, de que o investimento em tecnologia em geral e em instrumentos computadorizados para coleta de dados pode fazer avançar o ensino e a pesquisa em Psicologia e Análise do Comportamento. A ênfase do capítulo ficará com a discussão da interface entre Psicologia e Informática. Essa discussão se dará através da apresentação e análises críticas de *softwares* desenvolvidos para pesquisa e aplicações em Psicologia. As análises serão baseadas tanto nos relatos dos autores dos *softwares* quanto em avaliações realizadas pelo presente autor que manipulou alguns dos *softwares* e analisou os artigos que os descrevem. As descrições – ainda que de cunho mais pessoal do que uma avaliação técnica criteriosa – tanto dos aspectos positivos quanto negativos desses *softwares* têm a função apontar as direções que a programação da nova versão do *software* ProgRef tomou.

*Pequeno histórico do desenvolvimento da informática*

Desde a antiguidade, os homens buscam o desenvolvimento da tecnologia para realização de cálculos de maneira automática. Os ábacos foram um dos primeiros instrumentos para realizar cálculos.

**Figura 2.1** Ábaco

**Fonte**: Extraída de: www.corbis.com.br. acesso em: 16 abr. 2010.

O ábaco, um dos modelos exibidos na Figura 2.1, é formado por uma moldura com linhas paralelas em seu interior (como uma escada) que representam unidades, dezenas, centenas etc. Em cada uma dessas linhas eram fixados alguns objetos, tais como pedras ou pedaços de madeiras. Dependendo da disposição dos objetos nas linhas os cálculos aritméticos poderiam ser realizados com facilidade. Atualmente o ábaco é utilizado como ferramenta pedagógica para ensino da matemática (Fernandes, 2003).

A busca por inovação tecnológica para realização de cálculos culminou, após a Revolução Francesa no séc. XVIII, com o desenvolvimento das Máquinas Diferenciais utilizadas para o cálculo de impostos (Kowaltowski, 1996).

**Figura 2.2 -** Máquina diferencial

**Fonte**: Extraída de: www2.uol.com.br/sciam/reportagens/a_ origem_da_computacao_2.html. Acesso em: 16 abr. 2010.

As Máquinas Diferenciais, representada por um dos modelos na figura 2.2, apresentavam uma grande complexidade mecânica de engrenagens e peças, difíceis de serem montadas com a tecnologia da época. No Séc. XX (década de 1930) ocorreu uma aceleração no desenvolvimento de máquinas automáticas de cálculo, coincidindo com a disponibilidade de dispositivos eletro-mecânicos (relês) e eletrônicos (válvulas). As agências militares, tanto

nos Estados Unidos da América quanto na Europa, tinham um grande interesse em desenvolver tecnologia para cálculos complexos e com alto grau de precisão. As agências militares mobilizaram esforços, financeiros e humanos (cientistas), para o desenvolvimento tecnológico (Kowaltowski, 1996).

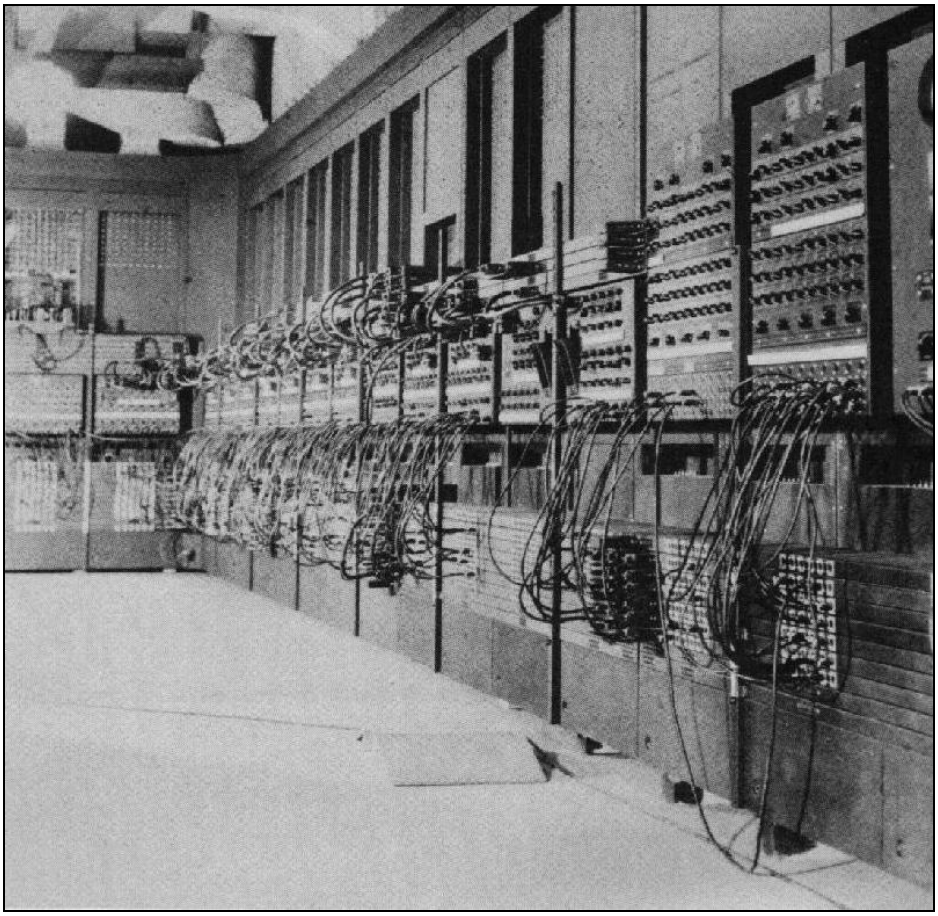
Para o desenvolvimento dos computadores era necessário não só o desenvolvimento tecnológico, mas uma mudança no que se entendia por "computador". O computador não deveria ser visto apenas como um instrumento puramente matemático, mas conceitualmente ser visto como uma máquina universal de processamento de informação. A influência teórica para essa mudança de perspectiva pode ser encontrada na teoria do matemático Alan Turing, que no final da década de 30 demonstrou ser possível usar uma máquina simples para executar uma imensa variedade de tarefas complexas e também que todo sistema formal pode, em princípio, ser automatizado (Campbell-Kelly, 2009).

Um sistema formal é composto de: (1) um conjunto de fórmulas e (2) um conjunto de regras de transformação para manipulá-las. Por exemplo, uma linguagem de programação é um sistema formal. Na década de 40, o matemático húngaro Jhon Von Neumann formalizou o projeto lógico de um computador. Esse projeto foi baseado em ideias e terminologias (e.g., o uso do termo "memória") oriundas da neurologia – particularmente o trabalho sobre operações lógicas no cérebro desenvolvido por Warren McCullough e Walter Pitts, ambos neurocientistas (Campbell-Kelly, 2009).

O desenvolvimento da computação pode ser visto em três áreas principais: (1) *hardware*; (2) arquitetura e (3) *software*. No *hardware*, o grande desenvolvimento tecnológico e o uso de circuitos integrados possibilitou um grande aumento na velocidade de processamento da informação. Quanto à arquitetura, o arranjo lógico de subsistemas que compõem um computador praticamente não evoluiu. Quase todas as máquinas em uso hoje em dia compartilham a mesma arquitetura básica do computador desenvolvido na década de 40. São computadores de "programa armazenado" (nome dado em referência à capacidade de armazenar na memória do computador – o *hard disk* ou HD, no caso – uma programação prévia, como os programas atuais) (Campbell-Kelly, 2009; Kowaltowski, 1996).

Os computadores existentes entre 1940 e 1950 eram difíceis de serem utilizados. A Figura 2.3 apresenta o ENIAC [*Electronic Numerical Integrator and Computer*]. O tamanho e as diversas conexões faziam com que os primeiros computadores tivessem como características negativas: alto custo, lentidão no processamento e cálculos poucos confiáveis e inexistência de *softwares* de apoio para a sua programação.

**Figura 2.3** - O computador ENIAC.



**Fonte:** Extraído de www.msp.gov.ec/dps/ morona_santiago/images/stories/eniac.jpg. Acesso em: 16 abr. 2010.

Toda a programação deveria ser feita em linguagem de baixo nível (também conhecida como programação de *hardware* ou pseudocódigo). O código do programa era ininteligível para humanos, formados basicamente de números binários (zeros e uns) ou decimais que correspondiam a uma instrução (e.g., o código 14 correspondia à função de adição). Esse tipo de programação era passível de erros e a tarefa de programação era muito difícil. Com as linguagens *Short Code* (e.g., UNIVAC I) a programação se tornou um pouco mais facilitada, pois era possível utilizar ferramentas de apoio para a programação. Isso possibilitava o uso de algumas palavras-chaves durante a programação, facilitando sensivelmente o entendimento do programa além de evitar erros. Mas isso tinha um custo, pois os programas eram em torno de 50 vezes mais lentos do que os desenvolvidos em linguagem de máquina (Sebesta, 2003).

Uma das primeiras linguagens de alto nível, linguagens com um código mais próximo das instruções utilizadas em uma língua (o inglês, no caso), foi a FORTRAN[12] desenvolvida pela IBM. Ela foi a primeira linguagem de alto nível a ser compilada. Os primeiros compiladores tinham pouco controle de erros, não facilitando a programação. Na época, os computadores ainda eram lentos, pouco confiáveis e utilizados principalmente em meios acadêmicos. A linguagem LISP surgiu, na década de 50, a partir da demanda da Inteligência Artificial, principalmente influenciados pela Linguística, Psicologia e Matemática: (1) os linguistas estavam interessados no estudo de linguagens naturais; (2) os psicólogos buscavam modelar processos mentais (e.g., aprendizagem e memória, armazenagem e a recuperação de informações humanas); e (3) os matemáticos queriam mecanizar certos processos inteligentes (e.g., demonstração de teoremas). Era preciso uma linguagem que fosse capaz de trabalhar com listas[13], principal característica da linguagem LISP. O ponto importante a ser ressaltado aqui é que o desenvolvimento da LISP exemplifica como problemas práticos de outras disciplinas cinetíficas influenciaram no desenvolvimento das linguagens de programação (Sebesta, 2003).

Com os computadores se tornando mais populares, no final da década de 70 e início da década de 80, foi desenvolvida a linguagem BASIC que atendia os requisitos dos microcomputadores da época por exigir pouca memória. Além de ser uma linguagem de fácil aprendizagem para os não-cientistas. O BASIC sofreu grandes críticas e caiu no esquecimento, ressurgindo na década de 90 com seu uso no Visual Basic®. O Visual Basic® foi projetado para desenvolver sistemas de *software* que tivessem interfaces com o usuário providas de janelas (Sebesta, 2003).

A programação em janelas se desenvolveu enormemente com o conceito de programação orientada a objetos (POO). Antes as linguagens eram procedurais, isto é, o código era dividido em procedimentos (ou sub-rotinas). Com esse método, erros de valores em determinados procedimentos eram prováveis de ocorrer principalmente em códigos com muitas linhas de programação. A manutenção do código-fonte era dificultada, pois uma pequena alteração em uma das linhas tornava impossível prever o impacto global no código-fonte. As linguagens de programação orientada a objeto sanaram alguns desses problemas das linguagens procedurais. Linguagens orientada a objeto tornam o código mais consistente e

---

[12] O uso da Linguagem FORTRAN para o desenvolvimento de um programa para o calculo do *quarter life* (índice para análise de desempenho do responder em FI) pode ser encontrado em Maurisse (1978). Neste artigo, o autor disponibiliza o código fonte da sub-rotina de cálculo do *quarter life*.

[13] A discussão sobre o conceito de listas em programação seria longa e desnecessária para os objetivos desse trabalho. Para o leitor interessado consultar: Sebesta (2003) ou acessar o sítio: http://www.dca.fee.unicamp.br/courses/EA072/lisp9596/Lisp9596.html (acessado em: 15/05/2010).

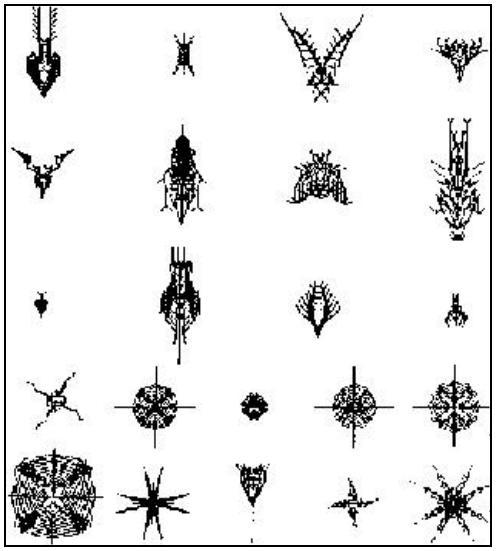
facilitam a sua manutenção. A sua característica principal é o conceito de herança, a capacidade que permite aos desenvolvedores de *software* a reutilização do código e dados. Não só a programação alterou, mas também os ambientes de desenvolvimento de *software*. A interface com o ambiente (a tela vista pelo usuário) é altamente gráfica, fazendo muito uso de janelas sobrepostas e menus suspensos (*pop-up*) e de um dispositivo de entrada (*mouse*) que facilita a interação com o ambiente de programação (Camara, 2006; Sebesta, 2003). Com os atuais ambientes de programação orientada a objeto é possível notar a busca por ferramentas que facilitem a programação. Por exemplo, com o Delphi® 5 é possível criar uma aplicação de banco de dados sem que seja necessário ao programador digitar uma única linha de código.

### *Intercambio entre a informática e as ciências*

Pelo breve histórico evolutivo das linguagens de programação nota-se uma preocupação para democratizar o uso de uma linguagem de programação. Muitas linguagens foram desenvolvidas de acordo com uma demanda. Pesquisadores das mais diversas disciplinas científicas podem escolher, dentre as muitas linguagens, a que melhor atenda a sua necessidade e a que mais oferece facilidades para sua utilização. Em diversas áreas científicas os pesquisadores podem utilizar da tecnologia digital para testar hipóteses, conhecer melhor teorias, analisar e correlacionar resultados, entre outros usos na produção de material científico.

Por exemplo, o biólogo Dawkins (2001) descreveu o desenvolvimento e a utilização de seu programa de computador “Biomórfo”. Esse *software* tinha por objetivo simular o processo de seleção cumulativa com o uso de uma representação virtual de um ser, o biomórfo. Em sua primeira geração, ele era representado por uma linha na tela do computador. Esse biomórfo, na versão inicial do *software*, era composto de nove genes cada um responsável por uma característica (por exemplo, o Gene 9 controlava a angulação da ramificação do biomórfo). Esses biomórfos eram capazes de se reproduzirem e seus genes sofriam mutações, a cada geração era apresentada para o operador do computador uma sequência de diferentes seres virtuais. O operador na função de “agente seletivo” escolhia o biomórfo que teria “sucesso reprodutivo”. Essa ação era capaz de produzir diferentes seres e demonstrar como a seleção cumulativa, partindo de algo simples como uma linha, poderia gerar figuras abstratas com alto grau de complexidade, como pode ser visto na Figura 2.4.

**Figura 2.4** - Alguns exemplos dos biomórfos “selecionados” pelo *software* desenvolvido por Dawkins



**Fonte**: Extraído de Dawkins (2001, p. 474).

Dawkins (2001) relatou que a sua experiência em programação foi muito importante para o sucesso do programa. Inicialmente, desenvolveu o programa em BASIC, mas características dessa linguagem na época tornavam o *software* lento, motivo pelo qual optou por outra linguagem, o Pascal. Ele continuou atualizando o *software* e ampliando as variáveis para manipulação, por exemplo, colocando genes responsáveis pela cor dos biomórfos. O sucesso do programa biomórfo exemplifica como um experimentador da Biologia Evolutiva utilizou da tecnologia digital para realizar suas pesquisas e representar uma teoria por meio de uma simulação em computador.

A computação também teve grande participação no desenvolvimento da teoria do Caos. A teoria do Caos descreve a complexa realidade em que vivemos através de modelos matemáticos e com uma linguagem própria que usa elegantemente termos como fractais e bifurcações, intermitências e periodicidades, difeomorfismos e mapas de intervalo. Com esse paradigma, pode-se dizer que a matemática tornou-se uma ciência experimental, fortemente apoiada nas imagens gráficas fornecidas pelos computadores. O Caos tornou-se não só uma teoria, mas também um método, um modo de fazer ciência. A teoria do Caos necessita de uma grande capacidade de processamento e cálculo dos computadores e criou a sua própria técnica de usar os computadores. As mais diversas disciplinas científicas têm usado os modelos da teoria do Caos para descrever seus fenômenos, tais como a meteorologia, a economia, a química, a física, a dinâmica de populações etc. (Gleick, 1989).

Gleick (1989) apresentou os estudos iniciais e os empecilhos encontrados pelos cientistas. Naquela época (década de 1970) as necessidades computacionais da teoria do Caos estavam mais relacionadas com a velocidade e capacidade de processamento. Isso era muito mais um problema de *hardware* do que de *software*. Para outra fascinante área da computação, a Inteligência Artificial (IA), os grandes desafios, atualmente, estão no desenvolvimento de *softwares* capazes de simular procedimentos computacionais supostamente encontrados nos animais.

A Inteligência Artificial tem como premissa que as atividades que constituem a inteligência consciente são todas elas algum tipo de procedimento computacional. Um dos conceitos-chave em Inteligência Artificial é o conexionismo, também conhecido como Processamento Distribuído em Paralelo (PDP) ou Redes Neurais. Trata-se de uma abordagem que focaliza modelos baseados na organização espontânea de unidades que interagem entre si segundo um conjunto de regras simples. Através dessa interação é possível "fortalecer", ou seja, alterar os pesos das conexões interferindo assim na saída da rede. O fortalecimento das conexões equivale ao aprendizado da rede (Churchland, 2004).

Um exemplo do uso da Inteligência Artificial na Psicologia é o *software* CyberRat, que tem por objetivo simular as atividades de um animal de laboratório (um rato) em um ambiente virtual. É uma apresentação em vídeo totalmente interativa. O *software* possui um banco de dados com 1800 *clips* em vídeo contendo comportamentos de um rato (e.g., explorar a caixa experimental, focinhar, pressionar a barra, consumir as pelotas de comida etc.) e utiliza algoritmos estocásticos (algoritmos não determinísticos, normalmente é utilizado o termo aleatórios, pois sua lógica durante a execução não é linear) para controlar a sequência de apresentação desses *clips*, simulando o comportamento de um animal real (pois os algoritmos estocásticos possibilitam a criação de animações em tempo real dando a impressão que o comportamento do rato não é determinado, ou programado) respondendo de acordo com as variáveis que o experimentador manipula (e.g., liberação de pelotas de comida contingente a uma dada resposta do rato). O experimentador tem acesso ao biotério virtual, onde pode nomear os ratos. É possível realizar experimentos com diversos programas de reforço. Por exemplo, uma sequência aplicada com fins didáticos em cursos introdutórios à Análise Experimental do Comportamento é: ambientação à caixa experimental, modelagem, reforço contínuo, extinção, recondicionamento, programa de razão fixa. A cada sessão experimental é gerado um gráfico de registro cumulativo. Esses registros são levemente diferentes dependendo do rato utilizado e do grau de privação (nota-se inclusive alterações comportamentais correlacionadas a saciação durante a sessão), demonstrando que a

programação do *software* não é uma programação linear. Com o CyberRat é possível executar uma sessão que duraria uma hora em poucos segundos, utilizando o recurso de "Simulação Rápida", assim, um procedimento de discriminação de 30 sessões experimentais, podem ser desenvolvidas em poucos minutos (Eckerman, Vasconcelos & Gimenes, 2006)[14].

Além de desenvolvimento de ferramentas com um poder de programação maior (e.g., uso da Inteligência Artificial), os avanços tanto na programação quanto de estrutura possibilitaram o desenvolvimento de pesquisas em um ambiente virtual (e.g., a Internet ou Intranet). Atualmente tem crescido o número de pesquisas cujas coletas de dados ocorrem *online*. Quando comparados com os experimentos em laboratórios tradicionais eles apresentam algumas vantagens, tais como: (1) o procedimento experimental pode ser automatizado, assim reduzindo custos e a quantidade de tempo gasto gerenciando o experimento; (2) os experimentos *online* podem ser realizados em qualquer local (i.e., em em um arranjo ambiental menos restritivo que um laboratório), e pode incluir acesso 24 horas, podendo aumentar o conforto do participante; (3) a padronização ética pode ser mantida porque o experimento é disponível publicamente para crítica; (4) é possível atingir uma audiência específica (através de *mailing lists* ou *newsgroups*) atingindo assim o grupo de interesse dos participantes na Web (por exemplo, estudantes de graduação de universidades particulares ou homens com mais de 30 anos de idade e curso superior completo). Mas algumas desvantagens devem ser consideradas quando experimentos são realizados de forma *online*: (1) o ambiente é mais variável, incluindo ruído, luzes e aspectos técnicos dos equipamentos, embora efeitos dessa variabilidade, talvez possam ser reduzidos solicitando que os participantes do experimento realizem suas sessões em um determinado tipo de ambiente; (2) experimentos *online* são vulneráveis para submissões múltiplas, ou seja, a pessoa pode participar da pesquisa diversas vezes, mas talvez esse tipo de vulnerabilidade possa ser reduzido solicitando informações pessoais, usar senhas ou uma verificação de endereço de IP [*Internet Protocol*]; (3) podem ocorrer desvios na amostra final, pois somente participantes interessados e/ou motivados podem começar (auto-seleção) e completar o experimento. Além disso, as evidências apontam que nos experimentos *online* a evasão é maior do que das taxas encontradas nos experimentos em laboratório (Dandurand, Shultz & Onishi, 2008; Reips. 2002).

---

[14] Foge ao escopo do presente trabalho discutir os alcances e limites do uso de simuladores (como o *CyberRat* ou o *Sniffy* - Alloway, Wilson, Graham & Krames, 2000) no ensino de Análise Experimental do Comportamento. Para uma visão crítica do uso do *Sniffy* ver, por exemplo, Tomanari e Eckerman (2003).

Com o objetivo de comparar os resultados obtidos em experimentos *online* com os realizados em laboratório, Dandurand et al. (2008) manipularam a localização em que o experimento era realizado: no laboratório ou *online* através da Internet. Além da divisão do local de realização do experimento, também foram controladas as condições de aprendizagem: (1) Reforço: os participantes recebiam *feedback* sobre seu acerto ou erro; (2) Imitação: os participantes observavam a execução correta do problema; (3) Explícito: os participantes recebiam instruções de como resolver o problema. A tarefa a ser realizada era a resolução do problema de gizmo[15] que consiste em encontrar, com uso de três escalas, o gizmo (objeto) que seja mais pesado ou mais leve do que o resto dos elementos de um conjunto de 12 gizmos. Verificou-se que os participantes que assistiram a várias demonstrações diferentes de resolver o problema (Grupo Imitação) e os que receberam instruções (Grupo Explícito) superaram aqueles que tinham consequências explícitas (*feedback*) para resolver o problema. Os participantes *online* foram menos precisos do que os participantes que realizaram as sessões em laboratório. A taxa de abandono da tarefa *online* foi superior a taxa de abandono da tarefa realizada em laboratório, mas essa diferença desaparecia quando ocorria uma combinação prévia com o departamento que o participante pertencia no sentido de incentivar a participação no experimento.

Como apresentado, os avanços na área da informática possibilitou a construção de ferramentas para pesquisas nos mais diversos domínios do conhecimento científico. Uma inter-relação que tem se mostrado produtiva e possibilitado que diversas áreas científicas respondam suas questões. Não podendo ser diferente a Psicologia também utiliza essas ferramentas. Com o objetivo de exemplificar esse uso serão descritos alguns *softwares* utilizados em pesquisas básicas ou aplicadas.

### *Informática e Psicologia: pesquisa básica e aplicada em Análise Experimental do Comportamento*

Alguns pesquisadores da área da Psicologia desenvolveram seus próprios *softwares* para realizar suas pesquisas ou para intervir em alguma área aplicada. A seguir serão descritos alguns desses *softwares*. O MED-PC® é um *software* comercial, diferenciando dos demais, que têm por características serem desenvolvidos para atender a um problema de

[15] Para a visualização do experimento acesse: http://lnsclab.org/html/BallsWeightExperiment/PlayVersion/play.html (acessado em 16/04/2010).

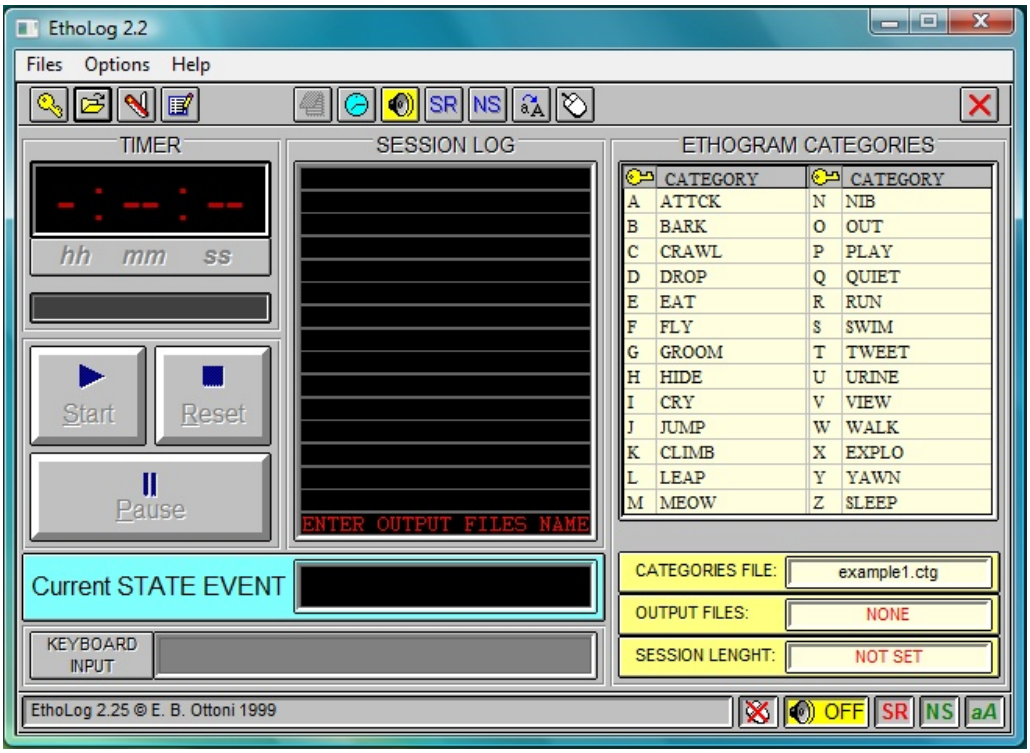
pesquisa da área do desenvolvedor. No entanto, como um dos objetivos do capítulo é discutir as alternativas que um Psicólogo possui para realizar sua coleta e análise de dados de modo informatizado, o MED-PC® se mostra como uma alternativa. Sua utilização também requer habilidades de programação do pesquisador (e.g., montagem de algoritmo e conhecimento de uma linguagem). Serão descritos alguns pontos sensíveis dos *softwares* baseados na descrição dos autores do *software*. Algumas descrições serão de cunho mais pessoal, baseado na manipulação de alguns dos *softwares* realizada pelo autor do presente trabalho. O objetivo foi verificar as possibilidades de arranjos experimentais e a interface com o usuário, com vistas em pontos aparentemente "positivos" (e que, eventualmente, foram utilizados na nova interface do ProgRef) ou "negativos" (e que, foram evitados na nova interface do ProgRef).

Etholog 2.2

O *software* EthoLog 2.2 (Ottoni, 2000) é uma ferramenta para a transcrição e cronometragem das sessões de observação do comportamento. Foi desenvolvido com o Visual Basic® e roda em ambiente Windows® (3.x/9.x). O autor utilizou métodos de programação para melhorar a acurácia do controle temporal. O controle temporal padrão do Visual Basic® tem precisão de centésimos de segundo e é afetado por outros eventos que estão ocorrendo no ambiente Windows®. Foi utilizada a rotina TIMERINFO da TOOLHELP.DLL que não é interrompida por eventos do sistema operacional Windows® e conta com uma precisão de décimos de segundos. Decisão importante do autor ao mencionar detalhes técnicos do desenvolvimento do *software*, pois serve para apresentar os cuidados quanto à programação que um desenvolvedor de *software* para coleta de dados deve se preocupar, além de auxiliar os que já estão programando a utilizar formas mais precisas de programação.

A instalação do *software* Etholog é simples. Ao abrir o programa, ver Figura 2.5, o usuário se depara com uma interface que poderia ser um pouco mais dedutível. Por exemplo, a lista de categorias comportamentais está visível (vazia) e quando o usuário tenta inserir dados tem a constatação que ela está bloqueada (poderia ficar invisível e no lugar dela um botão que abriria uma nova janela para inserção das categorias). Pontos sensíveis como esses poderiam ser evitados construindo uma janela para configuração da sessão experimental e outra para a sessão experimental propriamente dita. Um ponto positivo é a possibilidade de manter um banco de dados com diversas categorias, permitindo maior agilidade e menos trabalho para coletas de dados sucessivas.

**Figura 2.5** - Tela inicial do *software* Etholog 2.2 com a categoria *example* carregada



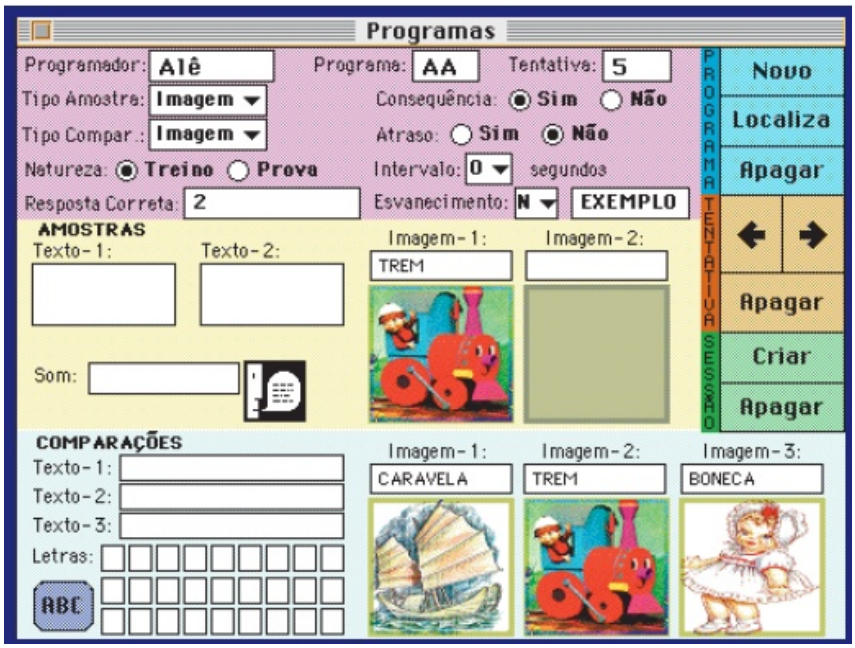

As categorias comportamentais e seus códigos essenciais são definidos pelo usuário e são salvos em um arquivo. Por exemplo, o código tecla <A> poderia equivaler à categoria “andar”. Portanto, quando a tecla <A> for pressionada o *software* registrará a ocorrência do comportamento de andar (e, eventualmente, sua duração – caso o usuário tenha programado o *software* para fazer esse registro). O EthoLog está concebido para lidar com dois tipos de eventos: (1) Eventos de Estado, que têm durações (e.g., duração do olhar em direção aos olhos de um entrevistador enquanto fala) e (2) Eventos Instantâneos, que não tem duração, mas vezes de ocorrência (e.g., número de passes durante um jogo de basquete). Quanto à categorização, o *software* apresenta uma grande flexibilidade, podendo ser geradas categorias para atender os mais diversos ambientes em que será realizada a observação.

Ao término da sessão são gerados arquivos de saída [.rep/.log] com um resumo da sessão de observação e com os dados na sequência completa dos comportamentos registrados e dados da cronometragem, que podem ser exportados para uma planilha eletrônica (e.g., Microsoft Excel®) para análises posteriores ou visualizados, editados e analisados (análise sequencial) no módulo *Toolpack* que compõe o pacote do Etholog.

*Mestre*

O estudo do fenômeno de equivalência de estímulos demanda um arranjo experimental complexo. Segundo Goyos (2004) um sistema informatizado para estudos na área de equivalência de estímulos evitaria erros humanos que poderiam ocorrer durante a coleta e controle experimental devido a grande complexidade das tarefas a serem executadas. Segundo o autor, outros benefícios que um sistema informatizado proporciona são: um maior rigor experimental e a possibilidade de aumentar os números de variáveis a serem estudadas (e.g., classes de equivalência com um número maior de estímulos).

**Figura 2.6** - Tela de programação da atividade do *software* Mestre




**Fonte**: Extraído da revista MacMania, v. 2, n. 15, p. 28, 1995.

Goyos (2004) desenvolveu o *software* Mestre[16] que permite a configuração de sessões de equivalência de estímulos, podendo ser utilizado tanto para fins de pesquisa, quanto para uma área aplicada, como o ensino e desenvolvimento de habilidades acadêmicas de alunos em séries pré-escolar, primeiro grau e na educação especial. O *software* Mestre® foi desenvolvido para rodar tanto em MAC quanto PC, ampliando o acesso e a facilidade no uso.

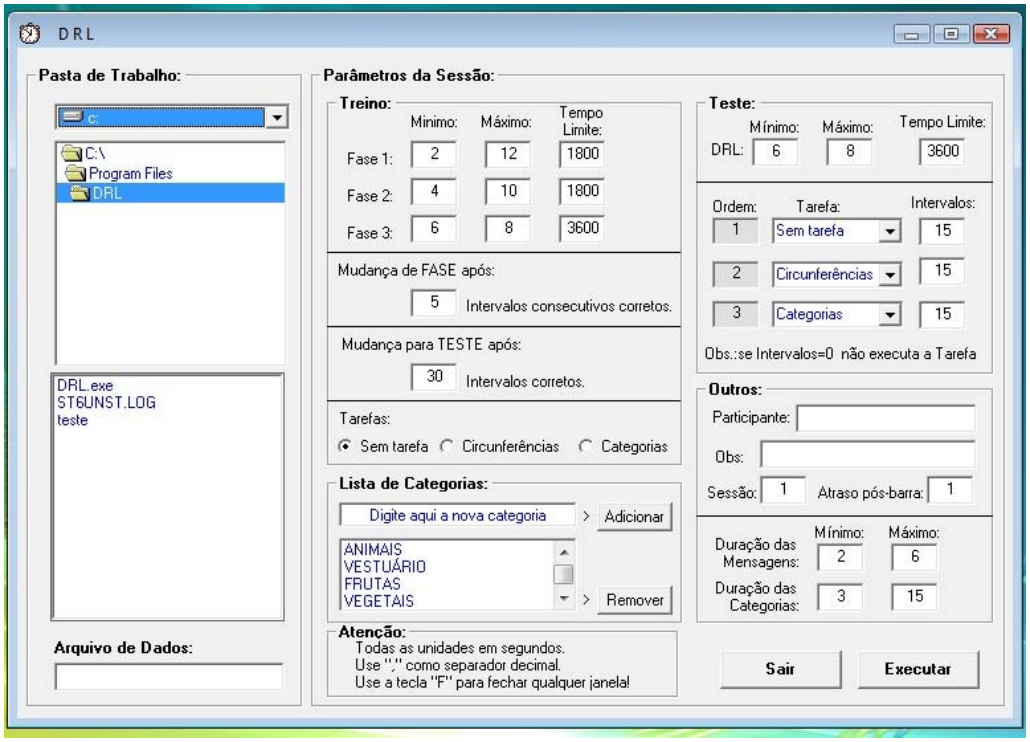
[16] Não há indicação da linguagem de programação utilizada no desenvolvimento do *software* Mestre. Uma descrição do *software* pode ser encontrada em: http://www.ufscar.br/~lahmiei/mestre.html (acessado em 16/04/2010).

A interface com o usuário é amigável, formada por figuras que representam as rotinas que seriam executadas. O usuário, ao criar uma nova tarefa, pode introduzir sons, imagens e textos tanto como estímulos modelos quanto como comparações. Existe certa flexibilidade do *software* que possibilita a inserção de novas figuras e sons de acordo com a necessidade da pesquisa ou da intervenção. O limite do *software* é de até dois estímulos como modelo e até três estímulos como comparação (Goyos & Almeida, 1994).

*Programa DRL*

O *software* DRL (Costa, Paula, Xavier & Bueno, 2007) foi desenvolvido para estudo experimental do conceito de julgamento temporal e possibilita a configuração da contingência de DRL [*Diferential Reinforcement of Low Rates*]. No programa de reforço DRL uma resposta é reforçada caso ela ocorra após uma passagem de tempo da resposta anterior, ou seja, o intervalo entre as respostas deve ser maior que t segundos. No *software*, a descrição mais adequada para o programa que pode ser configurado é um DRL com *Limited Hold*. Além do tempo do programa DRL está presente outra contingência, um intervalo de tempo máximo em que o participante deve emitir a resposta. Por exemplo, em um DRL 5 s com *Limited Hold* de 2 s, após a passagem de um intervalo de 5 segundos desde a última resposta, a consequência programada (uma mensagem escrita “correto”) esta disponível, mas só será liberada caso o participante emita a resposta em até 2 segundos após a disponibilidade, ou seja, o participante para emitir uma resposta “correta” (obter o reforço) deve responder com um intervalo entre respostas de 5 a 7 segundos. O participante recebia mensagens a cada resposta, sendo que: (1) se o intervalo entre respostas fosse menor que o valor de DRL a mensagem era “CURTO”; (2) se o intervalo entre respostas fosse maior ou igual que o valor do DRL e menor que o valor do DRL somado com o *Limited Hold* a mensagem era “CORRETO”; e (3) se o intervalo entre respostas fosse maior que a soma dos valores do DRL e *Limited Hold* a mensagem era: “LONGO”. O programa foi desenvolvido com o Visual Basic 6.0® e roda em ambiente Windows (98/2000/Me/XP).

**Figura 2.7** - Tela inicial do *software* DRL.




O *software* apresenta em apenas uma janela toda a configuração da sessão como apresentado na Figura 2.7. Isso possibilita ao pesquisador visualizar todos os valores escolhidos antes de executar a sessão experimental, diminuindo as chances de erros. No entanto, da forma como a tela de configuração da sessão está montada, os dados ficam misturados, há muita informação em apenas uma janela. O *software* não possibilita que a configuração da sessão seja salva para um uso posterior, muito comum em pesquisas em que se tem a mesma configuração para diversos participantes.

Mesmo assim, o *software* tem uma interface amigável a um usuário neófito, requerendo apenas algumas poucas instruções (e.g., o rótulo “Arquivo de dados” poderia ter um complemento ou ter outro conteúdo que indicasse que ali deve ser colocado o nome do arquivo de resultados e a listagem do diretório poderia conter um filtro para apenas os arquivos de resultados anteriores fossem visíveis e não qualquer tipo de arquivo). Tanto no *software* DRL quanto no Etholog o usuário deve entrar com um número significativo de dados para a configuração da sessão. No Etholog é utilizada uma lógica de janelas para inserção de dados e no *software* DRL apenas uma janela com muita informação fica disponível ao usuário. Além disso, como apresentado, o Etholog permite que os dados de configuração da sessão sejam salvos, enquanto que o *software* DRL não permite essa ação.

O *software* DRL foi desenvolvido para ter uma grande flexibilidade. Ele permite a edição de diversos parâmetros: (1) fase de treino; (2) número de respostas corretas para mudança de fase; (3) sinalização de realização de tarefas concorrentes; (4) configuração da lista de categorias, caso seja escolhida a tarefa concorrente de listar elementos pertencentes à categoria (e.g., categoria "Animais", o participante deve escrever em uma folha nome de animais: cachorro, pato, gato etc.); (5) apresentação de mensagens que sinalizam se o intervalo entre respostas está curto, correto ou longo, dependendo dos parâmetros configurados para o DRL; entre outras. Durante a sessão experimental o participante tem disponível um botão (*operandum*) e recebe como *feedback* as mensagens: "curto", "correto" ou "longo", dependendo do intervalo entre respostas. No final da sessão é gerado um arquivo que é facilmente exportado para uma planilha eletrônica (e.g., Microsoft Excel®), o que facilita o tratamento dos dados para análises numéricas e/ou gráficas (Costa, Paula, Xavier & Bueno, 2007).

*Med-PC*

Além das linguagens descritas anteriormente, as linguagens de programação para propósitos gerais (e.g., Basic, C, Pascal etc.), existem as linguagens para propósitos específicos para controle experimental em pesquisas envolvendo comportamento operante (e.g., Med PC e SKED II). Essas linguagens apresentam um vocabulário próximo ao do utilizado por pesquisadores da área da Análise Experimental do Comportamento, facilitando a programação e uma leitura do código-fonte (Gollub, 1991).

O Med-PC® (Med Associates, 2003) é um *software* para controle experimental utilizando caixas de Skinner, sendo possível a configuração de diversos parâmetros da sessão experimental (e.g., controle dos estímulos luminosos e/ou sonoros, liberação de reforço etc.) e aquisição dos dados dos dispositivos da caixa (e.g., pressão à barra). O Med-PC® IV é compatível com sistemas operacionais Windows® 95, 98, 2000, XP e Windows NT®. A configuração do arranjo experimental é realizada através de uma "linguagem de programação" criada especificamente para o Med-PC®, o *MedState Notation*®. Segundo o fabricante, é uma linguagem com comandos simples, cujas funções são dedutíveis e de fácil aprendizado. O programador monta um algoritmo com os comandos da linguagem (e.g., ON, OFF, SHOW, ADD, SET etc.) possibilitando grande flexibilidade e controle dos componentes da caixa, estímulos e mecanismos de reforço, bem como os registros dos dados. Outra facilidade é a reutilização do procedimento já escrito, podendo

sofrer pequenas alterações de modo rápido e prático, sendo também intercambiável entre os experimentadores.

Outra facilidade para utilização do MED-PC é o *software* Trans IV, que faz parte do pacote e traduz e compila os procedimentos MedState Notation® em arquivos DLL que podem ser lidas e as sessões experimentais executadas pelo MED-PC®. Esta aplicação também serve como um editor de texto com um índice de ajuda detalhada para linguagem de programação MedState Notation®. O Trans IV possui um código de erros para auxiliar o programador, o tipo de erro é apresentado de forma clara. Configuração de *hardware* é usada para criar um arquivo de configuração que informa ao MED-PC® quantas caixas estão conectadas, quantas entradas e saídas estão disponíveis para cada caixa, e como elas são identificadas (Med Associates, 2003).

Alguns *sites* disponibilizam protocolos de pesquisa já escritos na linguagem específica para o MED-PC® (e.g., http://people.bu.edu/Fabio, Acessado em 16/04/2010), mas em número muito menor do que os materiais encontrados para as linguagens de propósitos gerais, que disponibilizam grande quantidade de material de ensino (e.g., tutoriais, apostilas, vídeos-aula etc.), códigos-fontes abertos, fóruns de discussão, entre outros auxílios para a programação (Gollub, 1991).

O MED-PC® IV permite que o usuário colete dados de até 16 caixas de teste com até 1 milhão de elementos de dados, por sessão. Uma caixa de teste único poderia ter até 80 entradas e 80 saídas. Gabinetes de Interface para 8 ou 16 módulos são disponíveis pelo fabricante. Módulos com: 8, 12, 16 ou 24 entradas/saídas em várias combinações também estão disponíveis (Med Associates, 2003).

### *A escolha da linguagem de programação*

Gollub (1991) avaliou alguns critérios que deveriam influenciar na escolha de uma linguagem de programação por um pesquisador. As linguagens de programação para propósitos gerais apresentam como vantagens: (1) ambientes de desenvolvimentos fáceis de serem utilizados e pacotes completos (contendo editor, *debbug* e gerador de executável); (2) facilidade de encontrar ajuda de outros programadores; (3) flexibilidade, podendo o pesquisador desenvolver *softwares* para controle dos mais variados arranjos experimentais; (4) portabilidade, pode ser projetado para rodar em diversos computadores ou sistemas operacionais; (5) suporte para programação, tutoriais, fóruns de discussão etc. As desvantagens seriam: (1) empreendimento de tempo e esforço para aprendizagem da

linguagem e lógica de programação; (2) suporte de ajuda pode ser exíguo no caso de algumas linguagens menos populares; (3) programas longos e complexos podem esconder falhas tanto no controle experimental quanto no registro dos dados. Isso pode acontecer com qualquer linguagem, mas é esperado que ocorra com maior probabilidade com uma linguagem pouco familiar.

As vantagens para escolha de uma linguagem de programação de propósitos específicos, segundo Gollub (1991) são: (1) as construções lógicas do programa e o vocabulário da linguagem são próximos aos utilizados pela Análise Experimental do Comportamento; (2) ensinar a linguagem para o estudante pode fortalecer seu entendimento de procedimentos experimentais; (3) algumas dessas linguagens permitem que seja inserido sub rotinas de linguagens de programação de propósitos gerais, aumentando consideravelmente a flexibilidade. As desvantagens em utilizar linguagens de programação de propósitos específicos seriam: (1) pouca portabilidade para outros computadores; (2) usualmente requer uma interface de *hardware* específica; (3) algumas linguagens não permitem a inclusão de sub rotinas, limitando as contingencias que podem ser programadas; (4) dificuldade para encontrar ajuda de outros programadores.

A escolha de uma linguagem parecer ser um ponto fundamental para definir se o pesquisador terá sucesso ou fracasso no desenvolvimento de um *software* para pesquisa. Linguagens de propósitos gerais apresentam como principal característica a capacidade de resolver diversos problemas, cabendo apenas ao programador desenvolver uma lógica que permita a sua solução. No entanto, muitas vezes essa tarefa pode ser difícil e o pesquisador deve despender um grande tempo para o seu aprendizado. As linguagens de propósitos específicos têm como principal objetivo facilitar a sintaxe da programação, mas para arranjos complexos o pesquisador terá que montar suas rotinas com linguagens de propósitos gerais. O estudo de uma linguagem de propósitos gerais, aparentemente, se mostra mais satisfatório e proporciona ao pesquisador a capacidade de planejar experimentos sem a preocupação com os recursos limitados do equipamento.

Quando a questão é tempo, o empenho do pesquisador no aprendizado de uma linguagem não está focado no aprendizado da sintaxe da linguagem ou das palavras-chave. Aprender a montar algoritmos é requisito para qualquer linguagem de programação seja para propósitos gerais ou específicos, diferenciando apenas o ambiente de desenvolvimento (e.g., editor e compilador). A lógica de programação é o grande desafio e requer tempo (anos de programação) para que alguém desenvolva um raciocínio adequado e eficiente, implicando em um programa bem estruturado e com clareza na escrita do código-

fonte. A vantagem das linguagens de programação como Visual Basic®, nesse caso, é um grande acervo de arquivos de ajuda *on-line* e de códigos-fontes que serviriam como modelos para solução dos problemas encontrados durante a programação. Por exemplo, o programador que utiliza uma linguagem de propósitos gerais (e.g., Visual Basic®) para desenvolver um *software* para estudo de tempo de reação, pode ter como problema a precisão no cálculo da passagem de tempo entre um evento e outro. Discussão a respeito desse problema é facilmente encontrada na *internet*: fóruns de discussão, arquivos com códigos-fonte que servem como modelos etc.

O uso de uma linguagem de propósito geral parece facilitar a programação de um *software* mais generalista, podendo alterar muitas variáveis sem a necessidade de alterar a programação. A coleta de dados e a mudança das varáveis de controle podem, facilmente, ser realizadas por um pesquisador que não conheça de programação. Por exemplo, no *software* DRL, descrito anteriormente, o pesquisador tem as opções para manipular os parâmetros do programa DRL em valor mínimo, valor máximo e tempo limite. A alteração desses valores é realizada por caixas de texto padrão do Windows® tarefa bem intuitiva para a maioria dos usuários. Com a utilização do Med-PC® essa tarefa não seria tão simples e requereria o conhecimento da linguagem por quem irá coletar os dados e a cada alteração das variáveis seria como um novo *software* desenvolvido.

A Psicologia tem se beneficiado com o uso da informática para coleta de dados em suas pesquisas e também como ferramenta para intervenção em ambientes aplicados. O experimentador com experiência em programação em linguagens de propósitos gerais tem algumas vantagens. Mas essas vantagens requerem que os desenvolvedores pensem em *softwares* com vários recursos iniciais, códigos-fonte fáceis de serem mantidos e com facilidade de “complementação”.

No próximo capítulo serão discutidos os alcances e limitações do ProgRef v3. A partir dessa análise crítica será possível entender alguns pontos sensíveis no projeto do ProgRef v3 e como ou porque eles foram contornados no desenvolvimento do ProgRef v4.

# CAPÍTULO 3

## ALCANCES E LIMITAÇÕES DO *SOFTWARE* PROGREF V3

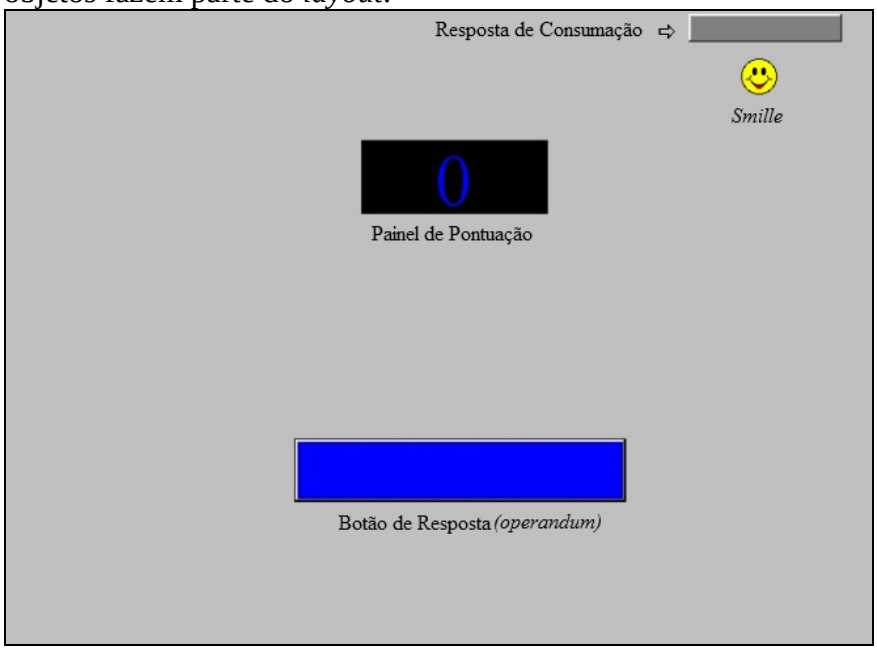
Neste capítulo será realizada uma análise crítica do *software* ProgRef v3 (Costa & Banaco, 2002; 2003). Em um primeiro momento serão discutidos os resultados acadêmicos propiciados pelo *software,* apresentando a produção científica proporcionada pelo ProgRef v3, através da descrição de algumas pesquisas desenvolvidas com esse *software*. Em seguida, serão levantados os aspectos técnicos do desenvolvimento do *software* ProgRef v3. Isso servirá para discussões sobre o desenvolvimento do ProgRef v4.

O *software* ProgRef v3 permite ao experimentador desenvolver pesquisas utilizando programas de reforço com humanos. Foi desenvolvido em Visual Basic® 6.0. Os requisitos mínimos para rodar o programa são: Windows® 95; processador 486; 8 MB de memória RAM; 4 MB de espaço livre no *hard disk* (HD); teclado e *mouse* padrão; monitor preto e branco ou em cores. Essa configuração foi almejada por permitir o uso de computadores considerados “obsoletos” e que poderiam ganhar utilidade nos laboratórios de pesquisa (Costa, 2006). A interface com o usuário é amigável e dedutível, apresenta um esquema de janelas que através de passos sucessivos permite a programação da sessão experimental. Requer apenas que o usuário tenha um domínio básico de utilização de *softwares* em ambiente Windows® e algum conhecimento sobre programas de reforço.

Com o ProgRef v3 é possível realizar sessões experimentais com programas de reforço simples ou complexos. Os programas disponíveis são: Reforço Contínuo (CRF); Razão Fixa (FR); Razão Variável (VR); Intervalo Fixo (FI); Intervalo Variável (VI); Extinção (EXT); Tempo Fixo (FT); Tempo Variável (VT); Reforço Diferencial de Baixas Taxas (DRL).

A Figura 3.1 exibe um *layout* da tela do *software* com a qual o participante interage durante as sessões experimentais.

**Figura 3.1** - *Layout* de uma tela da sessão experimental do ProgRef v3. Os textos explicativos não aparecem para o participante durante a sessão, apenas os objetos fazem parte do *layout*.




Em uma configuração mais utilizada, na tela da sessão fica visível para o participante o botão de resposta (*operandum*) e o painel de pontuação. O botão de respostas pode ser omitido da tela caso o experimentador decida usar a barra de espaço do teclado como botão de respostas. O visor de pontuação também pode ser omitido, mas esse detalhe da configuração será descrito mais adiante. A exigência ou não de uma resposta de consumação [*consumatory response*] é programada pelo experimentador. Se ela for exigida, o botão de resposta consumação aparecerá na tela como na Figura 3.1.; se ela não foi exigida nem o botão da resposta de consumação nem o *smile* aparecerão durante a sessão experimental. A possibilidade de poder configurar uma resposta de consumação é importante?

Um estudo que avaliou os efeitos da exigência ou não de uma resposta de consumação sobre o desempenho operante com humanos foi o de Matthews, Shimoff, Catania & Sagvolden (1977, Experimento II). Dez universitários foram expostos a um programa de reforço em FI. Após 25 reforços em CRF, seis participantes tiveram uma curta exposição (50 reforços) em VR antes de a contingência ser alterada para um FI 60 s e quatro estudantes foram expostos ao FI, cujo intervalo aumentou progressivamente até atingir um FI 50 s. Diferentemente da maioria dos estudos sobre programas de reforço com humanos, os autores modelaram a resposta de pressão ao botão em vez de estabelecê-la por instrução e criaram uma resposta de consumação: a cada ponto liberado os participantes tinham de parar de

responder no botão de resposta (o *operandum*) e deviam pressionar outro botão (o botão de resposta de consumação) para que o ponto fosse creditado no contador. Os participantes recebiam U$ 0,01 para cada ponto ganho na sessão. Este procedimento está mais próximo daqueles utilizados com não-humanos do que os estudos comumente realizados com humanos nos quais a resposta operante é estabelecida por instrução e nenhuma resposta de consumação, que interrompa o comportamento em andamento, são exigidos. Os resultados indicaram que quando uma contingência de FI foi posta em vigor os participantes tenderam a exibir um padrão de respostas em taxa baixa e alguns padrões de *scallop* foram observados. Ao final da sessão (90 minutos) o desempenho dos participantes era de uma resposta por intervalo. Esse padrão comportamental, embora não seja típico de não-humanos sob FI, sugere que o desempenho era sensível ao parâmetro temporal do programa de reforço. Os autores argumentaram que, possivelmente, diferenças no procedimento entre os estudos com humanos e não-humanos são responsáveis pelas discrepâncias nos resultados dos estudos sobre programas de reforço.

No *software* ProgRef v3 a resposta de consumação pode ser descrita da seguinte maneira: quando o critério para o cumprimento da contingência de reforço em vigor é cumprida, aparece no canto superior direito do monitor um ícone identificado comumente como um "*smile*". O participante deve, então, clicar com a seta do *mouse* sobre o botão que se localiza no canto superior direito da tela (botão da resposta de consumação), logo acima do *smile*. Ao fazer isso, o *smile* desaparece e o ponto é creditado no contador (painel de pontuação). Os *smiles* não são cumulativos. Enquanto um *smile* está presente no monitor, não aparecerá outro até que o participante pressione o botão da resposta de consumação. Depois que o *smile* desaparece – e o ponto é creditado – caso o participante volte a pressionar o botão de respostas (*operandum*), pode ganhar mais pontos. Caso o participante continue a pressionar o botão de respostas após o aparecimento do *smile*, as respostas são registradas, mas caso a contingência seja satisfeita o participante não receberá esse ponto. Essa programação aproxima-se de algumas configurações experimentais com ratos em que água é utilizada como evento consequente. Caso a resposta produza uma gota de água, mas o rato continue a pressionar a barra sem beber a gota de água, o próximo acionamento do bebedouro não produzirá uma gota de água "adicional". Algo semelhante ocorre em configurações experimentais com pombos em que o comedouro fica disponível por certo período de tempo após o cumprimento da contingência de reforço. Apesar de os *smiles* não acumularem, as respostas que eventualmente ocorrem após o aparecimento do *smile* são contadas para o cumprimento da contingência e a liberação do ponto seguinte. Por exemplo, em um FR 40,

após emitida a 40ª resposta o *smile* aparece na tela; se o participante não emitir a resposta de consumação e continuar pressionando o botão de respostas mais 10 vezes e, neste ponto, pressionar o botão de resposta de consumação, o *smile* irá desaparecer e o ponto será creditado no contador e, após a emissão de mais 30 respostas no *operandum*, outro *smile* aparecerá.

É possível programar a relação entre a resposta que completa a exigência do programa de reforço, o aparecimento do *smile*, a emissão da resposta de consumação e o crédito dos pontos de três maneiras diferentes. Na primeira configuração possível, os cronômetros que controlam a execução do *software* continuam em operação durante todo o experimento, ou seja, o tempo consumido pelo participante para pressionar o botão de resposta de consumação é computado como parte do intervalo entre reforços (IRI). Nesse caso, o intervalo de um FI, por exemplo, é iniciado a partir do aparecimento do *smile* (liberação do "reforço"), e não a partir da emissão da resposta de consumação. Na segunda configuração possível, quando o *smile* aparece, o botão de resposta desaparece, mas os cronômetros que registram os intervalos de tempo continuam em funcionamento (como descrito anteriormente). Esta contingência apenas impede que o participante emita respostas de pressão ao botão após a liberação e antes do "consumo" do reforço (i.e., o participante só poderá executar uma nova resposta operante – a resposta-alvo estudada – após pressionar o botão de resposta de consumação). Finalmente, na terceira configuração possível, quando o *smile* aparece, o *operandum* desaparece e os cronômetros que registram os intervalos de tempo são interrompidos. Assim que o participante clicar sobre o botão de resposta de consumação, o *operandum* reaparece e os cronômetros voltam a funcionar. Adotando-se esta opção o tempo gasto pelo participante para emitir a resposta de consumação não é somado com a pausa pós-reforço (nas opções descritas anteriormente esse tempo é somado).

Um estudo que investigou o papel da resposta de consumação e das instruções no desempenho de humanos em FI, utilizando do *software* ProgRef v3 foi o de Costa, Patsko e Becker (2007). A configuração da resposta de consumação utilizada foi como a primeira descrita anteriormente. No primeiro experimento, seis universitários foram expostos a um FI por três sessões de 20 minutos. Para metade dos participantes não foi exigida uma resposta de consumação para que pontos fossem creditados ao contador e para a outra metade, além da resposta de consumação os participantes receberam uma instrução "adicional" que descrevia como o participante deveria se comportar quando a resposta de consumação ficava disponível. Os resultados não indicaram um efeito claro da resposta de consumação. No segundo experimento, oito universitários foram expostos a um FI por três

sessões de 30 minutos. Os participantes dos dois grupos receberam a mesma instrução mínima sobre a tarefa experimental. O desempenho de três dos quatro participantes do grupo com resposta de consumação pareceu ficar sob controle do FI, apenas um participante apresentou o padrão de alta taxa de respostas. Esse padrão foi observado em três dos quatro participantes do grupo sem resposta de consumação. Os resultados sugerem que a interação entre a exigência de uma resposta de consumação e o tipo de instrução pode favorecer baixas taxas de respostas em FI com humanos.

Em suma, pesquisas (e.g., Costa, Patsko & Becker, 2007; Matthews et al., 1977; Raia, Shillingford, Miller Jr. & Baier, 2000) têm sugerido que quando a resposta de consumação é utilizada no delineamento experimental dos programas de reforço com participantes humanos, é mais provável verificar padrões de pausa pós-reforço, taxas de respostas mais baixas em programas FI (em relação a exposição ao FI sem a resposta de consumação) ou taxas de respostas maiores em VR do que em VI.

Uma vez que resultados de pesquisas têm sugerido que a exigência de uma resposta de consumação é importante nos estudos do comportamento operante com humanos, pode parecer sensato programar uma nova versão do *software* em que a configuração de uma resposta de consumação não fosse opcional, mas obrigatória. Mas esse não é o caso. Há situações em que o planejamento experimental pode "pedir" a retirada da resposta de consumação. Por exemplo, pesquisas feitas por Weiner (1964; 1965; 1969) sobre história comportamental com humanos nunca utilizou resposta de consumação. A replicação de alguns de seus resultados, como por exemplo, a persistência de um padrão comportamental de altas taxas de respostas em um FI-custo, após uma história de exposição ao FR, pode residir na exigência ou não de uma resposta de consumação.

Weiner (1965) realizou um estudo no qual seis participantes, pagos por hora, foram distribuídos em três grupos. Em uma primeira etapa, os participantes de um grupo foram submetidos a dez sessões de uma hora de FR 40; os participantes de outro grupo foram submetidos a 10 sessões de uma hora em FI 10 s e os participantes de outro grupo realizaram dez sessões de uma hora de DRL 20 s. Em uma segunda etapa, todos os participantes foram submetidos a dez sessões de uma hora em FI 10 s-custo. Os participantes submetidos ao FR 40 recebiam 100 pontos a cada 40ª resposta emitida; os participantes submetidos ao DRL 20 s recebiam 100 pontos para cada resposta com intervalo entre respostas maior do que 20 s (IRT > 20 s) e os participantes submetidos ao FI 10 s recebiam 100 pontos para a primeira resposta emitida após um intervalo de 10 s. Sob a contingência de FI 10 s-custo os participantes recebiam 100 pontos para a primeira resposta emitida após um intervalo de 10 s e perdiam um

ponto para cada resposta emitida durante o intervalo entre reforços. Sob cada programa de reforço um tipo distinto de cor de luz estava presente no painel no qual os participantes trabalhavam. Os participantes com história de FI (sem custo) responderam de forma constante e estável durante os períodos inter-reforços e diminuíram significativamente a taxa de respostas quando foram submetidos à contingência de FI-custo. Os participantes com história de FR emitiram alta taxa de respostas enquanto que aqueles com história de responder sob DRL emitiram baixa taxa de respostas sob o FI-custo.

O "custo" para um responder entre reforços em uma contingência de FI foi a perda de pontos que, como visto, não parecia ser superior a quantidade de pontos ganhos (i.e., ao ganho "líquido"). No experimento de Weiner (1965), aparentemente, o custo não foi alto o suficiente para provocar uma mudança substancial na taxa de respostas após a mudança nas contingências. O que aconteceria se em vez de perder um ponto para cada resposta emitida no intervalo entre reforços e ganhar 100 pontos por reforço os participantes perdessem, por exemplo, 10 pontos para cada resposta emitida durante o intervalo entre reforços e ganhassem 100 pontos por reforço?

Essa questão foi investigada em dois estudos que utilizaram o ProgRef. No delineamento proposto por Becker, Xavier, Soares, Banaco e Costa (2006) cinco universitários eram expostos à seguinte sequência: FR; FI-custo 1 (perda de um ponto para cada resposta emitida durante o intervalo entre reforços); novamente FR e, finalmente, FI-custo 10 (perda de dez pontos para cada resposta emitida durante o intervalo entre reforços). Os participantes recebiam 100 pontos a cada vez que as contingências de reforço programadas eram cumpridas. Assim como nas pesquisas de Weiner (1965; 1969; 1970), nas quais luzes de cores distintas estavam presentes em cada um dos programas de reforço utilizados, nessa pesquisa a cor do botão de respostas mudava com cada programa de reforço. Os participantes trocavam seus pontos por dinheiro ao final de cada sessão (R$ 0,03 por ponto). O estudo de Costa, Soares, Becker e Banaco (2006) foi idêntico ao de Becker et al., com exceção de que os participantes recebiam R$ 2,00 por sessão, independentemente do desempenho (i.e., os pontos não eram trocados por nada ao final das sessões). Nas duas pesquisas, durante as exposições ao FR, todos os participantes emitiram altas taxas de respostas. Quando expostos a contingência de FI-custo 1, apenas dois participantes (um do estudo de Becker et al., cujos pontos eram trocados por dinheiro e outro do estudo de Costa et al., que recebiam dinheiro por sessão) mantiveram taxas relativamente altas de respostas. Quando a contingência foi de FI-custo 10, todos os participantes, de ambos os estudos, emitiram baixas taxas de respostas.

Os resultados da pesquisa de Weiner (1965), quando comparados aos de Becker et al. (2006) e Costa et al. (2006), apontam para direções discrepantes. No caso das pesquisas de Weiner, foram observados efeitos da história de FR de longa duração (i.e., altas taxas de respostas durante a contingência de FI-custo). No caso da pesquisa de Becker et al. e de Costa et al., pareceu haver maior probabilidade de responder em baixas taxas quando as contingências de custo (tanto perda de um ponto quanto perda de 10 pontos) foram inseridas. Os resultados de Becker et al. e de Costa et al. também sugerem que o reforçador empregado não explica esta diferença, uma vez que no estudo de Costa et al. os participantes ganhavam dinheiro por sessão e pontos pelo desempenho (assim como os estudos de Weiner) e no estudo de Becker et al. os participantes ganhavam pontos trocados por dinheiro ao final da sessão e resultados semelhantes foram obtidos em ambos os estudos.

Uma possibilidade de explicação da discrepância nos resultados seria o uso de uma resposta de consumação. Weiner não utilizava uma resposta de consumação em seus experimentos ao passo que nos estudos de Becker et al. (2006) e Costa et al. (2006) sempre foi utilizada uma resposta de consumação. O efeito da resposta de consumação sobre o comportamento quando a contingência de reforço muda de razão para FI-custo ainda precisa ser investigado. Estudos investigando isoladamente essa variável ainda não foram realizados. No entanto, um *software* que é projetado e desenvolvido para possibilitar a escolha do pesquisador para essa variável (exigência ou não da resposta de consumação) é mais flexível e passível de responder mais questões experimentais, podendo ser utilizado por mais pesquisadores, sem a necessidade de uma nova programação (reestruturação do código-fonte).

Outra flexibilidade do *software*, citada anteriormente, é quanto ao painel de pontuação. O painel de pontuação pode ser configurado para ficar visível ou não. Deixando o painel de pontuação invisível, impossibilita que o participante tenha acesso aos pontos ganhos. Por exemplo, o experimentador poderia desejar programar uma sessão na qual o participante tivesse que emitir uma resposta de consumação, mas não tivesse acesso ao acúmulo de pontos durante a sessão experimental. Neste caso a única consequência por pressionar o botão de respostas seria o *smile* que apareceria toda vez que o critério estabelecido pelo programa de reforço fosse atingido. Esse procedimento pode ser mais semelhante aquele empregado com organismos não-humanos em que o reforçador (e.g., uma gota de água) é consumido, mas não pode ser acumulado.

Há também a possibilidade do experimentador optar por apresentar no painel um valor monetário em reais (R$) em vez de simplesmente pontos. Nesse caso, o experimentador configura um critério de conversão de pontos para reais. Por exemplo, R$

0,05 (cinco centavos de reais) a cada ponto, assim a cada vez que a contingência é cumprida pelo participante seria acrescido no painel R$ 0,05 que, no final da sessão, poderia ser trocado por dinheiro. Os efeitos das diferentes programações do visor de pontuação (ou sua ausência) sobre o comportamento humano em programas de reforço ainda não foram estudados.

Um exemplo da flexibilidade do *software* ProgRef v3 e sua utilização por pesquisadores de outros laboratório é o estudo realizado por Panetta, Hora e Benvenutti (2007) que investigaram a aquisição do comportamento "supersticioso": interações entre descrições de contingências e comportamento adquirido por relação acidental com reforço. Participaram do estudo quatro estudantes de Psicologia. Os participantes foram expostos a quatro sessões experimentais. As sessões eram compostas da apresentação de um programa de reforço durante 30 s seguidos de um período de *time out* de 10 s. O programa em vigor mudava a cada sessão. Na Sessão 1 um programa VR 6 estava em vigor. Durante a Sessão 2 um programa de extinção. Na Sessão 3 o programa era novamente o de extinção com mudança na instrução no início da sessão. Na Sessão 4 o programa em vigor era um programa de tempo variável (VT) 6 s, em média, a cada 6 s era apresentado um reforço ao participante, independentemente de seu comportamento. A instrução da Sessão 3 e 4 foi chamada de instrução incorreta, pois solicitava aos participantes que ganhassem o máximo de pontos possíveis. Ao final das sessões era solicitado ao participante que respondesse algumas questões que tinham por objetivo investigar se eles eram capazes de descrever as contingencias que estavam em vigor durante as sessões. Os resultados apresentados por três dos quatro participantes foi um aumento na frequência de respostas da terceira (EXT) para quarta (VT 6 s) sessão, mostrando que instruções que sugerem relação entre resposta e mudança ambiental podem facilitar a aquisição de comportamento “supersticioso”. Um aspecto interessante dessa pesquisa, para a discussão desse capítulo, é que o *software* pôde ser bem utilizado, gerando resultados relevantes, por pesquisadores que desconhecem linguagem de programação. O “pré-requisito” fundamental foi o domínio da Análise Experimental do Comportamento em geral e do tema de pesquisa em particular. O ponto importante é que parece que programar *softwares* de pesquisa que atenda características mais gerais de configuração, que vão além do interesse imediato do pesquisador-programador, pode ter impactos relevantes para a área de pesquisa como um todo.

O ProgRef v3 permite também a programação de uma contingência de *limited hold* (LH) para os programas FI, VI e DRL. O *limited hold* refere-se a um curto período de tempo no qual um reforço contingente a um programa de intervalo é mantido disponível, ao final do qual, uma resposta não produzirá a consequência programada até que

outro reforço tenha se tornado disponível pelo programa de reforço em vigor (Ferster & Skinner, 1957). Por exemplo, em um programa FI 24 s com LH 6 s os IRT's < 24 s ou IRT's > 30 s não produziriam *smiles* ou pontos, apenas os IRT's >= 24 s e IRT's <= 30 s seriam seguidos por esses eventos.

O experimentador pode realizar experimentos utilizando programas complexos (múltiplo ou misto). Neste caso um dos programas listados será escolhido para ser o Componente 1 e outro para ser o Componente 2. O *software* não permite o arranjo de programas complexos com mais de dois componentes. No caso de um programa de reforço múltiplo, a cor do botão de respostas mudará com cada componente do programa.

É possível habilitar a opção de "*carry over*" em programas complexos. Neste caso, os valores do final de um componente serão "conduzidos" para o início da próxima apresentação do componente. Por exemplo, em um múltiplo FR 30 – FI 10 s, durante a apresentação do Componente 1 (i.e., FR) o participante emite 20 respostas, ocorre a troca do componente e o Componente 2 (i.e., FI) entra em vigor normalmente. Quando o Componente 1 voltar a ficar ativo, o "*carry over*" transporta as 20 respostas emitidas anteriormente e o participante só precisará emitir mais 10 respostas (20 + 10 = 30) para cumprir a contingência de reforço. A mesma coisa acontece para o Componente 2, mas o que seria "carregado", nesse caso hipotético, seria o tempo transcorrido entre a última liberação do *smile* (ou pontos) e a mudança do componente.

Também é possível programar um período de "*time out*" entre os componentes. O *time out* (TO), suspensão discriminada das contingências de reforço, refere-se a um período de tempo sem reforço durante o qual o organismo, caracteristicamente, não se engaja no comportamento sendo estudado (ou pela extinção durante a apresentação um estímulo ou pela remoção da oportunidade de responder) e é utilizado como provas, marcadores em uma série de eventos, um método de eliminar os efeitos de um comportamento anterior etc. (Catania, 1998; Ferster & Skinner, 1957).

A importância do *time out* pode ser observada indiretamente em duas pesquisas realizadas com o *software* ProgRef. Costa, Soares e Ramos (submetido) fizeram uma replicação sistemática do estudo de Freeman e Lattal (1992). O objetivo geral foi verificar se estímulos presentes durante a fase de construção da história exerciam algum efeito sobre o comportamento em um programa de reforço subseqüente em humanos. Quatro universitárias foram expostas a um programa de reforço múltiplo FR-DRL cujo efeito foi avaliado sobre um múltiplo FI-FI subseqüente. Na primeira condição a cor do botão de respostas era diferente em cada componente e o número de respostas no FR foi ajustado de

maneira a aproximar a taxa de reforço nos dois componentes. Na exposição ao múltiplo FI-FI, as cores do botão foram aquelas da fase anterior. O intervalo do FI foi calculado com base no desempenho da fase anterior. Quando expostos ao múltiplo FR-DRL os participantes emitiram taxa alta de respostas no FR e taxa baixa no DRL. Quando o programa mudou para um múltiplo FI-FI, três das quatro participantes apresentaram alta taxa de respostas no FI cuja cor do botão foi correlacionada ao FR e baixa taxa de respostas no FI cuja cor do botão foi correlacionada ao DRL. Para uma participante, as taxas de respostas pareceram ficar sob controle do múltiplo FI-FI. Esses resultados foram diferentes daqueles de Freeman e Lattal que relataram efeitos da história de curta duração (i.e., após algumas sessões, o comportamento dos pombos pareceu ficar sob controle do parâmetro temporal da contingência de FI – o que ocorreu com apenas uma participante do estudo de Costa et al.).

Soares (2008) realizou um estudo semelhante ao de Costa et al., mas introduziu as seguintes modificações no procedimento: inserção de um *time out* entre componentes do programa múltiplo; utilização de um critério de estabilidade na taxa de respostas para mudança de uma fase para outra; a consequência programada (pontos trocados por dinheiro, em vez de apenas pontos) e aumento do número de sessões na fase de teste. Participaram quatro universitários que foram expostos a um programa múltiplo FR-DRL. A cor do botão de resposta era diferente para cada componente do programa múltiplo (verde para FR e vermelha para DRL). Em uma fase seguinte, os participantes foram expostos a um programa múltiplo FI-FI. Cada componente deste múltiplo era correlacionado a uma das cores do botão de resposta apresentadas anteriormente. Nas duas fases, um *time out* de 5 segundos ocorria entre os componentes do programa múltiplo. De maneira geral, os resultados replicaram aqueles obtidos por Freeman e Lattal. Observou-se que três dos quatro participantes apresentaram efeitos da história de exposição ao múltiplo FR-DRL somente por algumas sessões. Com a exposição continuada ao múltiplo FI-FI, o comportamento destes participantes tendeu a ficar sob controle da contingência presente. O comportamento de um dos participantes apresentou efeitos da história de exposição ao múltiplo FR-DRL durante toda a fase de teste (múltiplo FI-FI). Este participante foi exposto, então, a mais dez sessões de um múltiplo FI-FI, mas as cores dos botões de resposta foram trocadas para preto e branco. Observou-se que o comportamento deste participante mudou, ficando semelhante ao dos outros três quando foram expostos ao múltiplo FI-FI. Os resultados do presente estudo sugerem que o comportamento de humanos tende a ficar sob controle da nova contingência de reforço com a exposição continuada a esta contingência e, portanto, os efeitos da história são transitórios. Quando a resistência à mudança pareceu maior, a substituição dos estímulos –

cuja função havia sido selecionada durante a fase de construção da história (i.e., a mudança na cor do botão de respostas) – foi suficiente para produzir uma mudança comportamental e produzir um padrão que parecia sob controle da contingência presente.

A lista de valores para os programas variáveis (e.g., VI, VR, VT) pode ser gerada por três processos. No tipo sorteio, os números da lista são gerados de forma pseudo randômica, baseado no número de elementos, no valor médio e uma amplitude, todos parâmetros estipulados pelo experimentador. Depois de gerados os valores é possível alterar a ordem e/ou o valor de cada elemento dentro da lista. Outra forma de gerar a lista de valores é utilizando um dos dois tipos de distribuição estatística, denominadas: progressão de Fleshler e Hoffman (1962) ou pela progressão de Catania e Reynolds (1968). Essas distribuições geram valores pseudo-aleatórios com uma melhor distribuição, evitando que os valores se concentrem em um pequeno intervalo. O trabalho de Catania e Reynolds (1968) demonstrou que o comportamento de pombos era sensível a distribuição dos valores em um mesmo programa de VI. Isto é, um VI 30 s pode gerar diferenças na distribuição das respostas ao longo do tempo em função de como os intervalos foram programados para ocorrer. O ProgRef v3 mantém três possibilidades de gerar os valores de uma VI, além da possibilidade de entrar com os valores manualmente, caso o experimentador tenha decidido gerar os valores com a utilização de algum outro método. Essa flexibilidade deve ser mantida no ProgRef V4.

Para controle do encerramento da sessão o experimentador pode escolher o critério de tempo e/ou número de reforços, quando for realizar uma sessão de programa simples. Os arquivos de saída, ou seja, os relatórios e gráficos gerados podem ser salvos no diretório de escolha do experimentador. O experimentador pode solicitar que seja gerado um arquivo de resultado com o registro cumulativo das respostas do sujeito. Esse arquivo será gerado durante a sessão experimental em formato bitmap (.bmp) e pode ser facilmente editado em diversos editores de imagens, inclusive o *Paint*®, disponível em diversos computadores. Outro arquivo de saída é o que contém os tempos entre respostas (IRT, do inglês *interresponse time*). O IRT refere-se ao tempo decorrido entre duas respostas (Catania, 1998). Esse arquivo é gerado na extensão texto (.txt) sendo facilmente importado para o Excel® para que se realizem análises dos dados obtidos com auxilio de macros[17].

O *software* ProgRef v3 possui ainda alguns recursos adicionais, tais como: modelagem da resposta de pressão ao botão e sessões de treino. O experimentador pode optar

[17] Macro é uma série de ações registradas para executar os passos gravados pelo desenvolvedor (e.g. em ações repetidas no programa Excel®). Também é possível com a linguagem de programação do Microsoft Visual Basic for Applications® (VBA), poder criar uma macro personalizada (extraído de http://office.microsoft.com/pt-pt/excel/HA011189582070.aspx acessado em 20/04/2010).

por modelar a resposta de pressão ao botão em vez de fornecer uma instrução do procedimento a ser realizado. Nesse caso, não é possível utilizar o teclado para acionar o botão de resposta, somente utilizando o *mouse*. Na sessão de modelagem o *software* gera uma espécie de alvo virtual, invisível, ao participante. No centro desse alvo esta o botão de resposta, o *operandum*. Nesse alvo virtual existem gradações de uma determinada largura. A cada vez que o cursor do *mouse* passa por essas gradações em direção ao centro, ou seja, em direção ao botão um reforço é liberado. Ao atingir o botão e o participante executar a resposta operante (clique no botão) um programa de CRF entra imediatamente em vigor. O experimentador irá configurar o número de reforços, que servirá como critério para alterar o programa e o parâmetro do programa. Por exemplo, utilizando como critério 10 reforços, após a décima resposta em CRF, outro programa entra em vigor. O experimentador pode escolher entre: FR, FI, ou DRL. E até cinco valores para o programa escolhido, caso o participante atinja o critério de reforços configurado anteriormente o parâmetro é alterado para o seguinte. Na sessão de treino, a única diferença é que não há no início a liberação de pontos por aproximar o cursor do *mouse* do botão, apenas o incremento dos parâmetros a cada determinada quantidade de reforços.

A descrição dos parâmetros de configuração possíveis e a descrição de algumas das pesquisas realizadas, além de apontar algumas linhas de investigação que podem ser exploradas, demonstram a utilidade do *software* ProgRef v3. Entretanto, a versão 3 possui limites que foram contornadas com uma nova versão do *software*. A seguir serão apresentados alguns pontos críticos da versão ProgRef v3 que a versão 4 procurou evitar ou corrigir.

Inicialmente o ProgRef v3 foi desenvolvido – e almejava – o uso de computadores com baixo poder de processamento e, consequentemente, de fácil obtenção por se tratar de computadores obsoletos de baixo custo (ou até mesmo adquiridos através de doações). A configuração do *hardware* desses computadores suportava apenas os sistemas operacionais Windows® 95 ou 98. Essa medida se mostrou eficiente na época, pois foi possível ter a disposição um número de máquinas que davam certa fluidez à coleta de dados, otimizando o tempo a ser gasto em tal tarefa (além de manter máquinas sobressalentes caso alguma tivesse problemas). Com o avanço tecnológico (e interesses mercadológicos), novas versões do sistema operacional Windows® foram lançadas (e.g., XP, Vista e 7), sendo raro, ou quase inexistente, as que mantêm atualmente computadores com versões antigas, impossibilitando o uso do ProgRef v3.

A versão 4 supera essa dificuldade podendo ser rodado nas principais versões do Sistema Operacional Windows®. Talvez, no futuro, versões para Linux precisem

ser desenvolvidas, mas a utilização desse sistema ainda se encontra incipiente nas instituições de pesquisa e tende a gerar resistência[18] por parte dos usuários acostumados com o Window®. Discussões sobre a utilização de *software* livre em pesquisa deveriam ser fomentadas, pois ela não implica em uma mera substituição do sistema operacional (de Windows para Linux, por exemplo). Diversos *softwares* comerciais possuem mais ferramentas que o seu análogo com código aberto. Por exemplo, usuários avançados do Microsoft Excel® não encontram ferramentas com todos os recursos (e.g., Microsoft Visual Basic for Applications para desenvolver macros) e facilidades para desenvolvimento de planilhas, aumentando a resistência quanto à adoção do sistema livre.

Quanto à configuração da sessão experimental no ProgRef v3 a sequência de janelas para configuração dos parâmetros da sessão, que inicialmente foram desenvolvidas visando uma interface mais dedutiva, se mostrou pouco eficiente e antieconômica. Se um erro de configuração ocorre nas janelas iniciais e for apenas diagnosticado posteriormente, o usuário deve retornar até a janela que contém o erro, corrigi-lo e avançar novamente, não raro gerando confusão no pesquisador. Além disso, uma programação com janelas deste tipo é trabalhosa e arriscada, pois uma programação com esse recurso demanda grande concentração do programador e deve ser exaustivamente testada (Costa, 2006).

Outro ponto sensível na programação do ProgRef v3 é quanto aos arquivos de *output* (e.g., gráfico cumulativo, tabelas com frequência de resposta e IRT). Nessa versão o gráfico é gerado durante a sessão – há um processamento interno que vai construindo o gráfico, não visível ao usuário. Isso foi evitado na versão 4. Os recursos de processamento da máquina é mais bem otimizado. No ProgRef v4 o gráfico é construído após a sessão, facilitando e agilizando a sua configuração (e.g., inserção de rótulo nos eixos, espessura da linha, marcação dos reforços entre ouras). Essa alteração também irá facilitar o armazenamento dos dados do experimento e com uma maior flexibilidade de tratamento.

Talvez, como relatado por Costa (2006), um dos pontos críticos no desenvolvimento do ProgRef v3 foi a falta de planejamento. A programação foi realizada “em etapas” (i.e., os recursos foram sendo construídos à medida que seu desenvolvedor tinha uma ideia “nova” ou diferente). Isso comprometeu a lógica de programação e a estrutura do programa. A atualização e construção de novos recursos foi ficando cada vez mais difícil e

[18] Além de uma percepção pessoal por parte do autor, indicativos de dificuldades na utilização do Linux pode ser visto no artigo intitulado: “Empresas que adotam Linux ainda enfrentam resistência interna” disponível em: http://computerworld.uol.com.br/tecnologia/2009/05/21/empresas-que-adotam-linux-ainda-enfrentam-resistencia-interna/ (acessado em 16/04/2010)

com mais probabilidade de erros. A nova versão teve um cuidado especial com o planejamento, além de evitar dificuldades e erros na programação e, provavelmente, possibilitará facilidades para a construção de novos recursos do *software*, como por exemplo, alguns dos programas de reforço com características mais específicas.

O próximo capítulo discute alguns aspectos técnicos relativos à programação do ProgRef v4 (e.g., a programação orientada a objetos; a escolha da linguagem de programação); os recursos de hardware necessários para instalar e rodar o *software* e uma descrição dos recursos do ProgRef v4.

# CAPÍTULO 4

## DESCRIÇÃO DO *SOFTWARE* PROGREF V4

*Aspectos técnicos: plataforma, ambiente de desenvolvimento e linguagem de programação.*

Atualmente, os Sistemas Operacionais (SO) estão cada vez mais fáceis de usar, possuindo interfaces muito simples e atrativas visualmente. Simplificadamente, um SO administra todos os componentes de *software* e *hardware* de um computador (e.g., quando um botão do teclado é pressionado, o SO fará o controle e, se for o caso, mostrará em um editor de texto a letra ou símbolo correspondente à tecla pressionada) (Flynn & McHoes, 2002).

As facilidades e funcionalidades dos SO surgiram de forma escalonada, muitas vezes, dependentes de um desenvolvimento na área de *hardware* (e.g., elevado custo da memória RAM de alta capacidade). Por exemplo, a interface gráfica do Windows® Vista é resultado de mais de 20 anos de desenvolvimento, desde as primeiras versões deste SO. As principais versões do Windows® foram: 3.0; 3.11; 95; 98; Me; XP; Vista e a última versão lançada 7[19] (Gugik, 2009). Todas essas mudanças nos SO afetaram diretamente os desenvolvedores de *software*, pois a cada lançamento deveriam ser verificadas as compatibilidades e possíveis alterações que poderiam comprometer a precisão ou, até mesmo, a execução do *software*. Essa é uma das principais limitações do ProgRef v3, que somente é compatível com a versão do Windows® 95; 98 e Me. A nova versão foi projetada para ser compatível com as principais versões do SO Windows® (XP, Vista e Seven) sem perder de vista a acurácia e fidedignidade dos dados coletados.

O ProgRef v3 foi desenvolvido em uma programação procedural (PP) utilizando a ferramenta de desenvolvimento Visual Basic® 6. Uma programação procedural tem como principal característica a divisão do programa em procedimentos [*procedures*] – chamadas na programação BASIC de sub-rotinas. Durante a execução do *software* esses procedimentos (sub-rotinas) comunicam-se entre si (e.g., sub-rotina “gera_relatório” solicita um cálculo para outra sub-rotina “soma_total”). Quando um *software* exige poucas linhas de programação, isso não é um problema. Entretanto, à medida que a complexidade do *software* aumenta, cresce também o número de linhas de programação e de sub-rotinas. Isso dificulta a manutenção do *software*, além de comprometer o desempenho e a precisão dos dados.

[19] Até a data de edição do presente trabalho.

Segundo Camarra (2006), a programação procedural exige do programador um planejamento rigoroso da lógica de programação para a construção de um *software*, uma característica pouco comum em desenvolvedores neófitos (cf. Costa, 2006). Ao utilizar a lógica da programação procedural o desenvolvedor tende a criar programas de forma desorganizada e códigos nos quais o processamento segue um longo trajeto, passando por várias sub-rotinas. Essa característica implica não só nos possíveis *bugs* durante a execução do programa, como na dificuldade de manutenção e futuras atualizações.

Uma alternativa a PP é a linguagem de programação Orientada a Objetos (POO). Esta facilita a manutenção e o entendimento das linhas de código (Camarra, 2006). Essa "facilidade" de Programação Orientada a Objeto é relativa, lembre-se estamos comparando com a PP, e pode ser exemplificada por analogia com o ditado popular: "Não precisamos inventar a roda a todo o instante". Uma das principais características da POO é a possibilidade de reutilização do código-fonte do programa. Ao desenvolver um objeto (e.g., objeto "roda", via programação por linhas de código) ele pode ser reutilizado nos mais diversos programas, por exemplo, reutilizado nos "veículos", nos "caminhões", nos "tratores" e assim por diante, todos ele utilizam o objeto "roda", este objeto possui diversas propriedades, nesse caso: calibragem, diâmetro, cor, textura etc. Assim, dependendo de onde quisermos reutilizar o objeto "roda", basta alterar essas propriedades. No contexto do ProgRef v4 um objeto é, por exemplo, o "programa de reforço em Razão Fixa". Vários *softwares* poderiam reutilizar esse objeto desde que o objetivo fosse o desenvolvimento de uma contingencia de razão fixa. Por exemplo, com o objeto "razão fixa" seria possível construir um *software* para o estudo de programas concorrentes de reforço e bastaria acrescentar esse objeto ao *software* sem ter de programar toda a lógica da razão fixa novamente. Isso possibilita que novos programadores desenvolvam seus *softwares* não se preocupando com a lógica do controle dos programas de reforço, que na maioria das vezes é parte mais complexa.

Como apontado anteriormente, o ambiente de desenvolvimento do ProgRef v3 foi o Visual Basic® 6. Costa (2006) apontou que a familiaridade pessoal com a linguagem Basic® teve grande peso na escolha para o desenvolvimento e investimento de tempo para aprofundamentos na linguagem Visual Basic®. A ferramenta de desenvolvimento do ProgRef v4 foi o Visual Studio® 2008 (VS).

O Visual Studio® 2008 é uma ótima ferramenta para desenvolvimento de *softwares*. Seu ambiente de desenvolvimento é bem completo e de fácil manuseio. O VS 2008 permite o desenvolvimento de aplicações na plataforma .NET (.NET *Framework*) que, resumidamente, é uma camada de *software* para execução e desenvolvimento de aplicações

que fica acima do SO. Projetada para simplificar o desenvolvimento de aplicações em um ambiente altamente distribuído como a internet, a plataforma .NET aumenta a portabilidade do *software*, ou seja, a capacidade dele ser executado em diversos ambientes. A principal característica da plataforma .NET é a linguagem comum em tempo de execução [*Common Language Runtime*] (CLR). O objetivo do CLR é aumentar a confiabilidade e segurança em aplicações e reduzir o volume de repetição de código – que aumenta a probabilidade de erros – além de facilitar o acesso a recursos específicos do SO, ponto importante para o desenvolvimento de *software* para pesquisa em Psicologia, que exige um grau de precisão maior e obtido através de programação que acessam algumas funções do SO.

As diferenças entre o VB 6 e o VB.NET podem ser vistas no: (1) ambiente de desenvolvimento que foi redesenhado para abrigar todas as linguagens do VS (e.g., Visual Basic, C# e Visual C++) ; (2) na sintaxe da linguagem e nos modelos de classes e objetos. A linguagem foi modernizada, foram excluídas algumas palavras-chave como *GoTo*[20] e inseridas outras como *Inherit*, além de alterar o modo de outras palavras chaves como *Return e Integer*; (3) houve alteração também no comportamento em tempo de execução dos componentes compilados (Ian Oliver & Bond, 2001). Essa última alteração foi bastante significativa, pois era uma das principais críticas levantadas contra o VB quando comparadas com outras linguagens como C++.

A escolha da linguagem VB.NET facilitou a comunicação entre os pesquisadores envolvidos no presente projeto (pois não foi necessário o aprendizado de uma outra linguagem). Surgirá a possibilidade de parcerias e/ou divisões de tarefas no desenvolvimento de novos *softwares*, isso proporciona uma maior agilidade no desenvolvimento dos programas, algo crucial quando se tem um prazo para término do projeto de pesquisa.

O conhecimento e familiaridade com a linguagem VB também proporciona uma grande vantagem para o pesquisador, pois facilita que ele desenvolva macros com mais funcionalidades através de implementações na linguagem *Visual Basic for Applications* (VBA) disponível nos programas do pacote Microsoft Office® (e.g., Excel, Word, Acess entre outros) que contêm o programa de desenvolvimento *Microsoft Visual Basic*. Isso fornece ao experimentador uma enorme gama de facilidades. Por exemplo, ele pode programar uma

[20] *GoTo* é uma função utilizada em programação procedurais, se imaginarmos o programa procedural como um texto, cada linha é um comando. Inicia-se a execução na linha 1 e percorre todo o texto linha por linha. A função *GoTo* possibilita pular trechos ou voltar na leitura, daí a ideia de "vai para". Para familiarizar o leitor e voltar à discussão realizada a respeito das dificuldades encontradas na PP. Imagine a leitura de um texto que é cheio de notas referenciando outras páginas, creio que o leitor já se aborreceu com esse tipo de referencia, imagine a manutenção e o entendimento de uma lógica de programação construída dessa forma.

macro que realiza uma determinada análise de seus dados e gere um relatório. Após a coleta de dados basta um simples clique para o relatório ser gerado. Além da agilidade, outro ganho com a utilização de macros é o aumento da precisão das análises realizadas, pois diminui a probabilidade de erros humanos (e.g., na inserção de fórmulas; seleção de células a serem analisadas etc.).

Pode-se argumentar que o conhecimento e aprendizagem de uma linguagem de programação leve tempo e necessite o empenho do experimentador (cf. Costa, 2006). Soma-se a esse arranjo a cultura tecnológica dos cursos de Psicologia, que dificulta o desenvolvimento e até mesmo utilização de ferramentas computacionais para realização de pesquisas. Diante desse contexto é importante que o *software* ProgRef v4 atenda a essa demanda e foi construído para pesquisadores que não estão, necessariamente, familiarizados com o uso da tecnologia.

Os cientistas interessados em novas áreas de pesquisas, nas quais pela primazia ainda não se encontram equipamentos capazes realizar a coleta e/ou tratamento adequado dos dados, se deparam com uma dificuldade muito mais em relação ao *software* do que com *hardware*[21]. Muitos computadores domésticos apresentam características para rodar os mais diversos programas para pesquisa.

### *Especificação de hardware recomendada para rodar o ProgRef v4*

O *software* pode ser instalado em computadores com sistema operacional Windows®, preferencialmente em versões superiores ao Windows XP® SP3, compatíveis com a plataforma *Microsoft Framework*.NET 4.0®. Os requisitos de *hardware* são: configuração mínima recomendada, Pentium 1 GHz ou superior com 512 MB de RAM ou mais; espaço em disco livre de 50MB ou mais.

[21] Ao leitor interessado no desbravamento de uma nova área de conhecimento e como as dificuldades encontradas na coleta dos dados em pesquisas experimentais podem ser superadas por medidas de baixo custo. Ver Nicolelis (2011), nesta obra há um relato do desenvolvimento de um equipamento de estimulação facial de um roedor baseado em um motor elétrico vendido em *kits* para *hoobistas* e cotonetes. A criatividade se mostrou uma competência importante para o experimentador em questão.

*ProgRef v4: aspectos de utilização*

O *software* foi projetado visando um usuário com conhecimentos básicos na utilização de programas em ambiente Windows®. A interface com usuário foi projetada para ser amigável, dedutível e funcional.

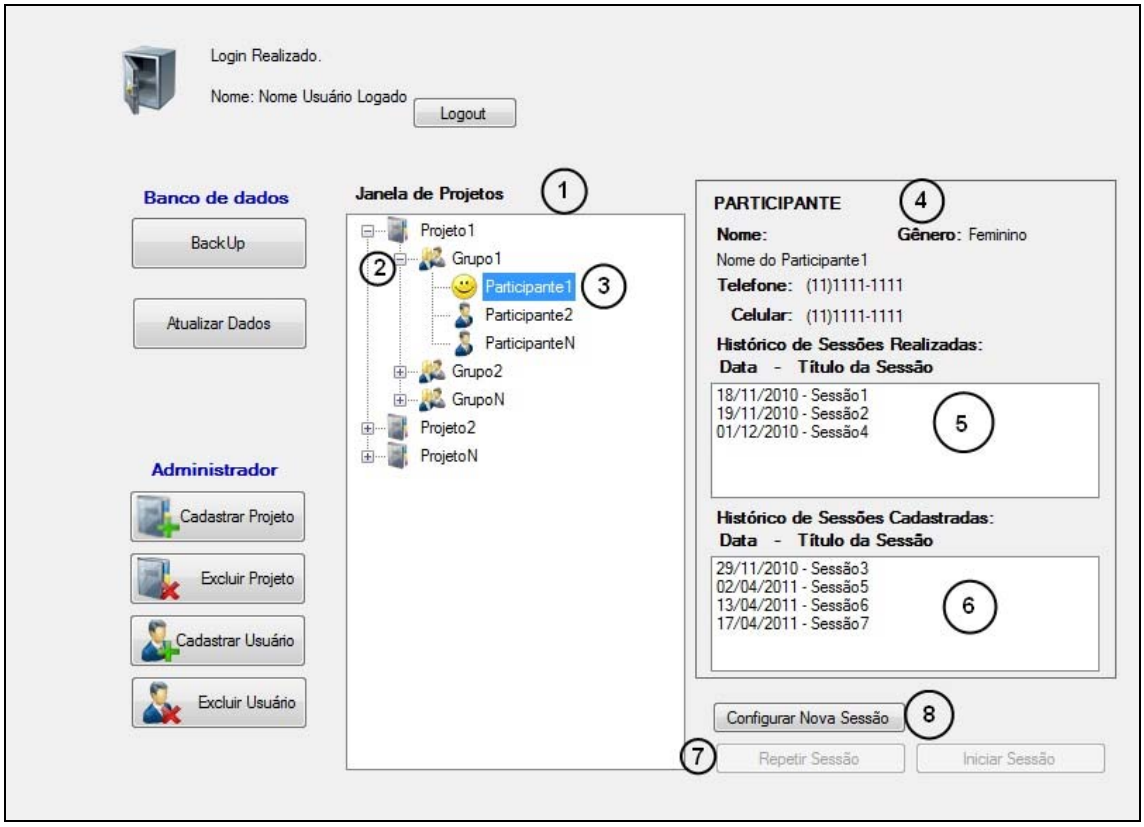
Para facilitar o gerenciamento dos projetos de pesquisa e aumentar a segurança dos dados experimentais coletados o *software* possui um banco de dados Microsoft Access® protegido por senha que pode ser acessado apenas pelos programas de coleta e análise do ProgRef v4. É necessário um cadastro dos usuários que podem ser categorizados como administrador ou experimentador. Aos administradores é possível o cadastro e exclusão de projetos de pesquisa. Também cabe ao administrador adicionar experimentadores e correlaciona-los aos projetos que terão acesso. O cadastro do projeto é composto pelos campos: título do projeto, agência financiadora e uma breve descrição dele. Para auxiliar na organização dos participantes no projeto, o experimentador pode distribui-los em grupos.

Para ter acesso às funcionalidades do programa o usuário deve realizar um *login*. Isso aumenta a segurança e principalmente o controle dos dados coletados. A Figura 4.1 exibe a tela inicial do ProgRef v4 após *login*. As informações sobre os projetos de pesquisa são apresentados como uma árvore de diretório “Janela de Projetos” (Marcador 1 da Figura 4.1) a raiz apresenta os projetos cadastrados, na figura nota-se os títulos “Projeto 1”, “Projeto 2” e “Projeto N”. Esses projetos estão disponíveis para o usuário logado. Os subdiretórios apresentam os grupos, no exemplo da figura “Grupo 1”, “Grupo 2” e “Grupo N” (Marcador 2 da Figura 4.1), todos pertencentes ao “Projeto 1”. Na sequência são apresentados os Participantes, no caso: “Participante 1”, “Participante 2” e “Participante N”(Marcador 3 da Figura 4.1).

Ao selecionar um dos itens da “Janela Projeto”, aparecerá o quadro ao lado com as informações cadastradas. No caso da Figura 4.1, o Marcador 4 mostra as informações do “Participante 1” , seus dados pessoais e as sessões realizadas (Marcador 5 da Figura 4.1) e as cadastradas (Marcador 6 da Figura 4.1). Caso o experimentador queira repetir o mesmo arranjo experimental de uma sessão já realizada, basta selecioná-la na lista “Histórico de Sessões Realizadas” e clicar no botão [Repetir Sessão] (Marcador 7 da Figura 4.1).. Será exibido um relatório com os dados da configuração da sessão experimental e o único campo que poderá ser alterado é o título da sessão, todos os demais parâmetros permanecerão com os mesmos valores. Na lista “Histórico de Sessões Cadastradas” (Marcador 6, Figura 4.1) são

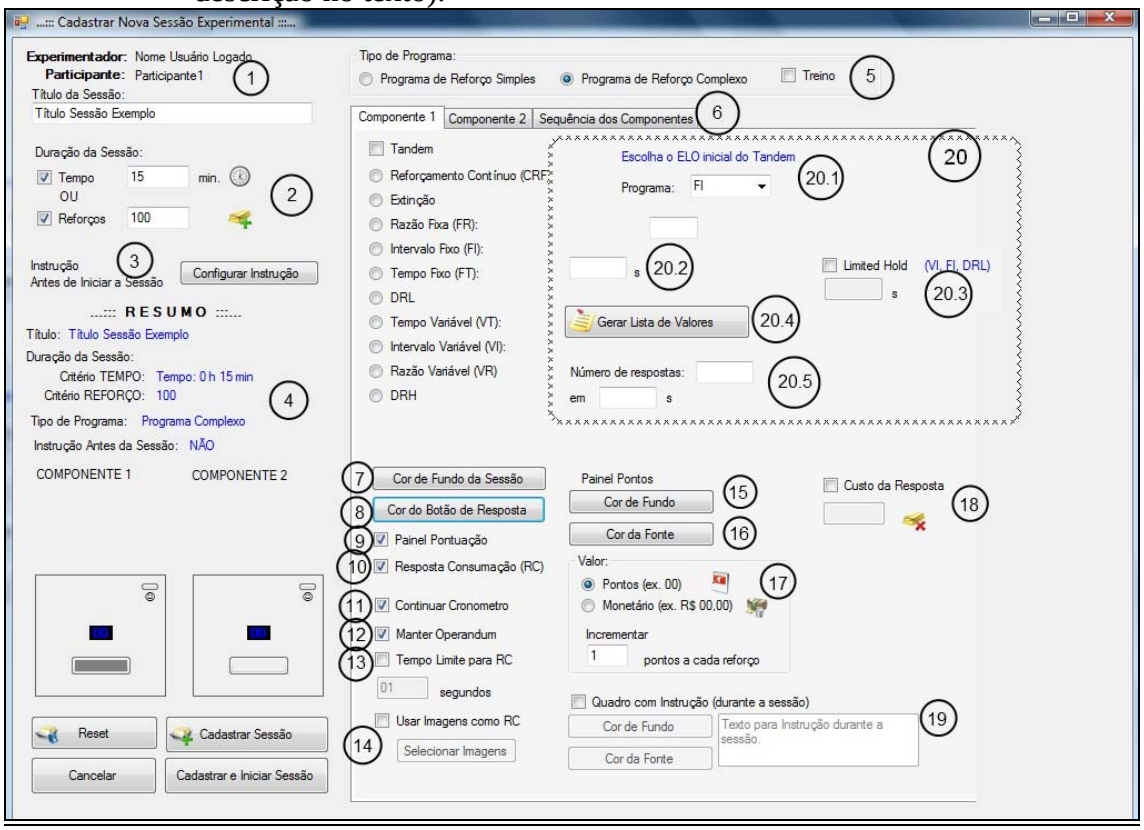
exibidas as sessões configuradas pelo experimentador mas que ainda não foram coletados os dados.

**Figura 4.1** - Tela inicial do ProgRef v4, após o *login* do usuário.




Ao clicar no botão [Configurar Nova Sessão] (Marcador 8 da Figura 4,1) um formulário para a configuração da sessão experimental como apresentado na Figura 4.2 irá abrir. O experimentador tem em apenas um formulário todas as possibilidades de configuração do arranjo experimental. O ProgRef v3 permite um grande números de arranjos de configurações das sessões experimentais, que foi mantido pelo ProgRef v4 atendendo os mais variados problemas de pesquisa. A nova versão tem como diferencial e incremento a possibilidade de utilização dos programas Tandem e DRH.

**Figura 4.2 -** Formulário de configuração da sessão experimental. Marcadores (ver descrição no texto).




Na parte superior esquerda da Figura 4.2 há um painel de informação com o nome do experimentador que esta logado e do participante que irá realizar a sessão experimental a ser configurada. Para facilitar a programação da sessão e evitar que o experimentador erre na inserção dos dados recomenda-se que ele inicie com os dados gerais da sessão: “Título da Sessão” (Marcador 1 da Figura 4.2), que auxilia no gerenciamento e controle do projeto de pesquisa e na busca dos dados para análise; “Duração da Sessão” (Marcador 2 da Figura 4.2) pode ser por tempo (unidade em minutos), ou por número de reforços. O controle de término da sessão feito pelo número de reforços não considera o custo de resposta e/ou a quantidade de pontos por reforços (i.e., a “magnitude” do reforço) (a configuração do painel pontos será descrita mais adiante). Caso o experimentador preencha os dois campos (tempo e reforços) a sessão será finalizada com o primeiro critério atingido.

É possível configurar uma instrução que ficará visível ao participante, antes do início de uma sessão experimental propriamente dita. O experimentador pode configurar a cor de fundo, a cor do texto e o texto da instrução clicando no botão [Configurar Instrução] (Marcador 3 da Figura 4.2). Caso o experimentador não configure uma instrução para ser exibida na tela apenas um botão [Iniciar Sessão] ficará disponível ao participante.

O formulário de configuração da sessão experimental possui uma grande quantidade parâmetros configuráveis, o que pode dificultar a visualização dos parâmetros da sessão que foram configurados. Para auxiliar a visualização da programação de uma sessão experimental há um resumo (Marcador 4 da Figura 4.2) que é atualizado a cada valor preenchido, inclusive com um *layout* da tela que será exibida durante a sessão experimental.

*Configuração dos Programas de Reforço*

Na parte superior do formulário o experimentador escolhe o tipo de programa de reforço que deverá vigorar durante a sessão experimental. A escolha é entre programas simples ou complexos, com a opção de treino (Marcador 5 da Figura 4.2). As abas das páginas (Marcador 6 da Figura 4.2) “Componente 1” e “Componente 2” são idênticas e contêm os parâmetros dos programas de reforço que podem ser manipulados na sessão experimental. Ao selecionar o Programa Simples a inscrição na aba será alterada de “Componente 1” para “Programa Simples” (e os parâmetros de configuração serão os mesmos que para um componente). Os programas de reforço disponíveis são: Reforçamento Contínuo (CRF); Extinção, Razão Fixa (FR), Intervalo Fixo (FI), Tempo Fixo (FT), Reforçamento-Diferencial-de-Baixas-Taxas (DRL), Tempo Variável (VT), Intervalo Variável (VI), Razão Variável (VR) e Reforçamento-Diferencial-de-Altas-Taxas (DRH). Para os programas FI, VI e DRL há a opção de configuração do *limited hold* (LH) ou contenção limitada. O LH refere-se a um curto período de tempo no qual o reforço ficará disponível, após o critério de tempo do programa de reforço vigente ter sido atingido. Por exemplo, em um arranjo experimental com a configuração de um programa FI 10 s LH 2s (lê-se, intervalo fixo de 10 segundos com *limited hold* de 2 segundos), o reforço ficará disponível após a passagem de 10 s, mas se o participante não emitir uma resposta em até 2 s, o reforço é cancelado (Ferster & Skinner, 1957). Portanto, o participante só ganharia pontos neste programa de reforço se a reposta ocorrer entre 10 e 12 segundos desde o último reforço obtido (ou desde o início de uma sessão), caso o participante pressione o botão no 13º segundo o reforço não é liberado e um novo intervalo é iniciado.

O experimentador pode manipular o *layout* da tela da sessão. Por exemplo, a cor de fundo da tela da sessão experimental pode ser configurada pelo botão [Cor de Fundo da Sessão] (Marcador 7 Figura 4.2) e a cor do *operandum* (o botão de respostas) pode ser

configurado pelo botão [Cor do Botão de Resposta]. Em ambos os casos será exibido ao experimentador uma paleta de cores para escolha.

O experimentador pode configurar se o “Painel de Pontos” (Marcador 9 da Figura 4.2) ficará ou não visível. Dependendo do problema de pesquisa pode ser interessante que o participante não tenha acesso ao total de pontos (reforços) obtidos. Neste caso, a única consequência por pressionar o botão de respostas e cumprir a contingência do programa de reforço em vigor seria a apresentação do *smile* (que, neste caso, deveria ter sido configurado – a configuração da resposta de consumação será descrita mais adiante). Se o Painel de Pontos for habilitado, há a possibilidade de configurar as formatações das cores de fundo e fonte através dos botões [Cor de Fundo] (Marcador 15 da Figura 4.2) e [Cor da Fonte] (Marcador 16 da Figura 4.2). Outra flexibilidade dos parâmetros de configuração do “Painel Pontos” é a conversão em unidade monetária, neste caso o símbolo “R$” aparecerá antes do valor de reforço. Pode-se configurar um incremento para apresentação dos reforços (Marcador 17 da Figura 4.2). Por exemplo, ao habilitar a conversão em unidades monetárias pode ser interessante ao experimentador valorar o reforço criando a seguinte regra: um reforço será igual a R$ 0,05 (cinco centavos) ou caso o tipo de reforço seja pontos a regra poderá ser que a cada reforço seja incrementado 10 pontos no painel de pontuação. No exemplo acima caso o participante tenha recebido 10 reforços o painel iria apresentar os valores, R$ 5,00 (cinco reais) ou 100 (pontos), a depender da unidade de reforço adotada (monetária ou pontos).

O painel pontos poderá exibir valores negativos se o custo da resposta tiver sido programado. O custo da resposta (Marcador 18 da Figura 4.2) refere-se à quantidade de pontos que será subtraída para cada resposta operante emitida. Por exemplo, em um FI 10 s, sem nenhum custo e 100 pontos por reforço o participante ganharia 100 pontos para a resposta emitida após a passagem de 10 s (desde o último reforçador liberado); se a contingência fosse um FI 10 s com custo 1 e 100 pontos por reforço, o participante ganharia 100 pontos para a resposta emitida após a passagem de 10 s, mas perderia um ponto para cada resposta emitida durante a sessão experimental (para exemplos de pesquisa com custo da resposta, ver Weiner, 1965; 1969; 1970 e Costa, Soares, Becker e Banaco, no prelo).

Outra flexibilidade do programa diz respeito à possibilidade de habilitar a Resposta de Consumação (RC) (Marcador 10 da Figura 4.2). Para uma apreciação da importância da resposta de consumação nas pesquisas operantes com humanos ver Costa, Patsko e Becker, 2005; Matthews et al., 1977; Raia, Shillingford, Miller Jr. & Baier, 2000). Os parâmetros de configuração no caso de uma RC ser exigida são: (1) “Continuar Cronômetro” (Marcador 11 da Figura 4.2), que possibilita que o tempo de apresentação da RC

não seja contabilizado para o término da sessão (seria como uma pausa da sessão); (2) “Manter Operandum” (Marcador 12 da Figura 4.2), que programa a retirada do *operandum* durante a RC,(neste caso o botão de respostas sumiria, ficando disponível apenas o botão da RC; (3) “Tempo Limite para RC” (Marcador 13 da Figura 2), que estabelece um critério temporal para limitar a disponibilidade do reforço (i.e., se a RC não ocorrer em determinado período de tempo após o aparecimento do *smile*, ele desaparecerá e o reforço será perdido; (4) “Usar imagem como RC” (Marcador 14 da Figura 4.2), apresenta na tela do computador uma imagem para o participante por um período de tempo, configurado pelo critério do Item 3 (“Tempo Limite para RC”). As imagens deverão estar em um diretório selecionado pelo experimentador. Os formatos de imagens aceitos pelo ProgRef v4 são: jpeg, jpg, gif, wmf, png e bmp.

Além da instrução exibida ao participante da pesquisa antes do início da sessão (programada no botão [Configurar Instrução], Marcador 3 da Figura 2, é possível configurar uma instrução para ficar disponível durante a sessão experimental. (Esta instrução ficará visível ao participante no canto superior esquerdo da tela – ver Marcador 1 da Figura 4.6 – e será descrita mais adiante). Os parâmetros de configuração dessa caixa de instrução são: (1) cor da fonte; (2) cor de fundo; (3) texto (Marcador 19 da Figura 4.2).

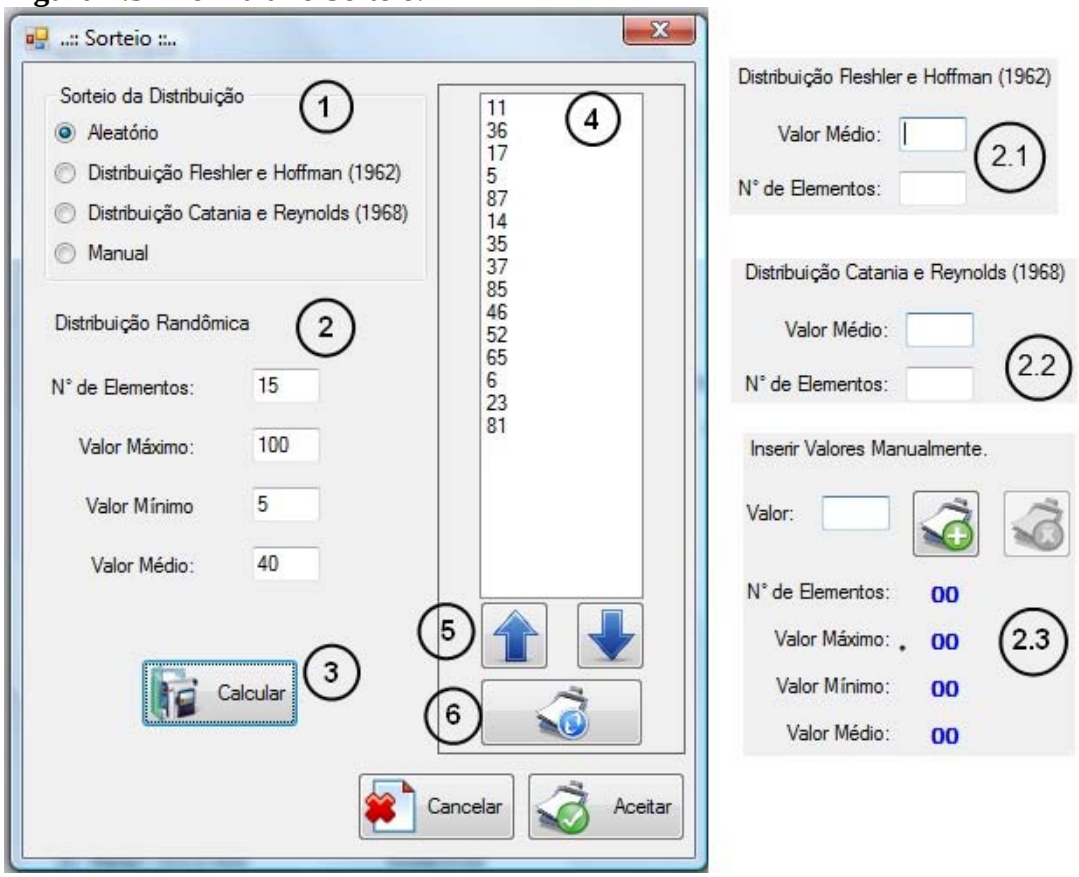
Os parâmetros apresentados no retângulo (Marcador 20 da Figura 4.2) refere-se aos parâmetros de configuração dos programas, a visualização dos itens depende da escolha do programa. Caso o experimentador escolha o programa Tandem, o Marcador 20.1 da Figura 4.2 ficará visível, para configuração do Elo inicial. O Marcador 20.2 da Figura 4.2 ficará disponível para edição do programas FR, FI, DRL ou FT. No caso de escolha de um dos programas FI, VI ou DRL, o Marcador 20.4 da Figura 4.2 também ficará visível para configuração do *Limited Hold.* Na escolha do programa DRH (Marcador 20.5 da Figura 4.2, a lógica “Número de respostas X em T s” fica disponível para que os valores X e T sejam configurados.

Para gerar a lista de valores dos programas variáveis (VR; VI e VT), o ProgRef v4 apresenta um formulário denominado “Sorteio” (Figura 4.3), que é aberto pelo botão [Gerar Lista de Valores] (Marcador 20.3 da Figura 4.2). O experimentador terá a disposição quatro modos para gerar a lista de valores a serem usadas como parâmetros destes programas de reforço, sendo três automáticos e um manual (Marcador 1 da Figura 4.3). A Figura 4.3 apresenta a seleção da opção “Aleatório”. Os parâmetros dessa opção podem ser vistos no Marcador 2 da Figura 4.3 e são: (1) número de elementos da lista; (2) valor máximo; (3) valor mínimo; (4) valor médio. O botão [Calcular] (Marcador 3 da Figura 4.3) gera a lista

de valores automaticamente que são exibidos em uma caixa de texto na parte direita deste formulário (Marcador 4 da Figura 4.3). Durante uma sessão experimental, o *software* utilizará os valores (para razão ou intervalo) na ordem em que eles aparecem nesta caixa de texto. Ao chegar no último valor o *software* recomeça a lista e a percorre até o término da sessão. As setas (Marcador 5 da Figura 4.3) permitem que o experimentador altere a ordem de apresentação de um determinado elemento da lista de valores. O botão abaixo das setas (Marcador 6 da Figura 4.3) altera a sequência dos elementos de modo randômico.

Além do sorteio da distribuição dos valores poder ser feiro pelo modo aleatório, descrito anteriormente, as outras duas formas de gerar valores automaticamente são através das distribuições estatísticas, denominadas de: (1) “Distribuição de Fleshler e Hoffman (1962)” (Marcador 2.1 da Figura 4.3); (2) “Distribuição de Catania e Reynolds (1968)” (Marcador 2.2 da Figura 4.3). Essas distribuições geram valores pseudo-aleatórios com uma melhor distribuição, evitando que os valores se concentrem em um pequeno intervalo. Por exemplo, um sorteio aleatório para um programa de VI 10 s, poderia gerar uma lista com valores muito altos e extremamente baixos (e.g., 1-1-1-1-1-3-20-20-20-21-21) e, mesmo assim, cumprir os critérios para um VI 10 s.

**Figura 4.3** - Formulário Sorteio.

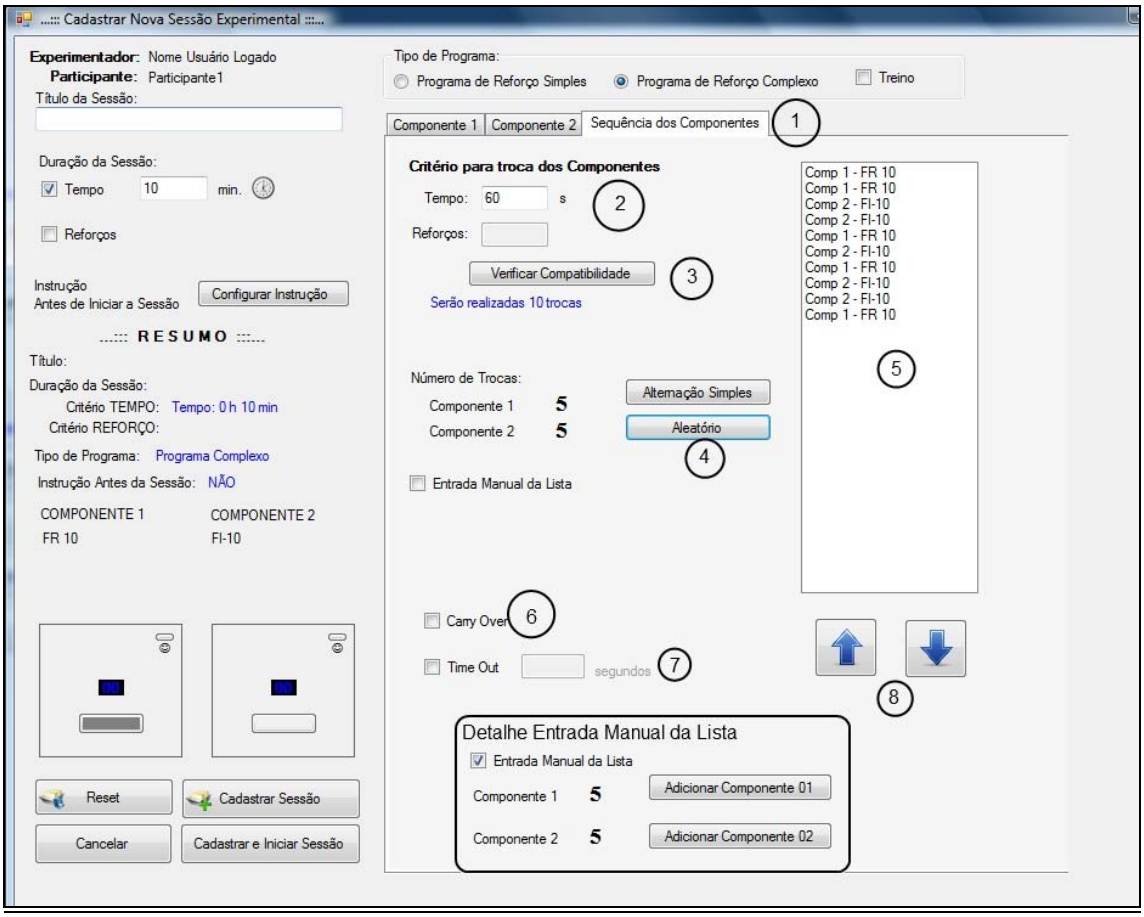


As Distribuições citadas anteriormente evitam esse tipo de sorteio, deixando os valores distribuídos de forma mais equitativa por todo o intervalo. Por exemplo, a distribuição de Fleshler e Hoffman (1962) de valor médio 10 e 11 elementos, gera os valores 1-1-3-4-5-7-9-11-15-20-34. Nota-se que os valores estão distribuídos mais uniformemente do que na lista anterior. O trabalho de Catania e Reynolds (1968) demonstrou que o comportamento de pombos era sensível a distribuição dos valores em um mesmo programa de VI. Isto é, um VI 30 s pode gerar diferenças na distribuição das respostas ao longo do tempo em função de como os intervalos foram programados para ocorrer. No ProgRef v4 essas duas progressões geram listas com valores ordenados de forma crescente. Após gerada a distribuição o experimentador pode utilizar as setas (Marcador 5 da Figura 4.3) ou o botão abaixo delas (Marcador 6 da Figura 4.3) para embaralhar a lista.

Caso o experimentador deseje configurar uma lista ou garantir que a mesma lista seja utilizada como critério por diversos experimentos, ele pode entrar com os valores manualmente (Marcador 2.3 da Figura 4.3). O experimentador insere o valor no campo indicado e clica no botão de inserção de valores, representado com o sinal de [+]. O botão com sinal [x] retira um valor, previamente selecionado, da lista. As informações do número de elementos e dos valores máximo, mínimo e médio são atualizadas após cada operação de inserção ou retirada de valor.

No arranjo experimental em que o “Tipo Programa” escolhido foi o “Programa de Reforço Complexo” (ver Marcador 5 na Figura 4.2), o experimentador deverá configurar a sequência de apresentação de cada componente. Os parâmetros são inseridos no formulário apresentado na aba “Sequência dos Componentes” (Marcador 1 da Figura 4.4). A quantidade de trocas dos componentes é baseada pelo critério de término da sessão, o botão [Verificar Compatibilidade] (Marcador 3 da Figura 4.4) realiza o cálculo de compatibilidade para que os componentes sejam apresentados em igual número de vezes. Por exemplo, uma sessão com duração de 10 min e critério de troca dos componentes 1 min será compatível, pois cada componente entrará 5 vezes em vigor durante a sessão. Pode ser o caso do experimentador configurar o término da sessão pelo número de reforços ou por um determinado período de tempo, o que acontecer primeiro. Nesse caso ele deve escolher um dos parâmetros (tempo ou reforço) como referência para o cálculo de quantidade das trocas, mas o término da sessão pode ocorrer em qualquer um dos casos configurados podendo deixar o número de apresentação dos componentes diferente.

**Figura 4.4** - Configuração da sequencia dos componentes de um programa complexo.




Escolhido um critério para troca, e este sendo compatível com a escolha de duração da sessão, será habilitado as formas de distribuição da sequência. O modo aleatório (acionado pela pressão ao botão [Aleatório], Marcador 4 da Figura 4.4) faz uma distribuição pseudo-randômica: os valores são sorteados, mas um filtro impossibilita uma sequência maior que três vezes do mesmo componente. A lista apresentada no Marcador 5 da Figura 4.4 foi gerada por esse método. Outra possibilidade automática de geração da sequência de componentes é através do botão [Alternação Simples]. Neste caso a sequência terá os Componentes 1 e 2 alternados a cada apresentação.

Na parte inferior da Figura 4.4 são exibidos os controles para inserção manual dos componentes na lista de sequência. Esses controles são exibidos ao se selecionar a opção “Entrada Manual da Lista”. (Os controles não aparecem nesta posição na tela do ProgRef v4. Este quadro foi adicionado na parte inferior da Figura 4.4. para facilitar a visualização de todos os controles desta aba). Na frente dos rótulos “Componente 1” e “Componente 2” é apresentado o número de vezes que cada componente pode ser inserido na

lista. Os botões [Adicionar Componente 1] e [Adicionar Componente 2] insere os valores na lista (na caixa de texto do Marcador 5 da Figura 4.4) obedecendo o critério de não permitir mais de três vezes consecutivas o mesmo componente.

Independente de ter gerado a lista de sequência dos componentes de forma automática (aleatória ou alternação simples) ou manual, o experimentador pode alterar a ordem de apresentação dos componentes utilizando as setas [⇧] ou [⇩] (Marcador 8 da Figura 4.4). Também neste caso, será respeitado o critério de, no máximo, três componentes iguais na sequência.

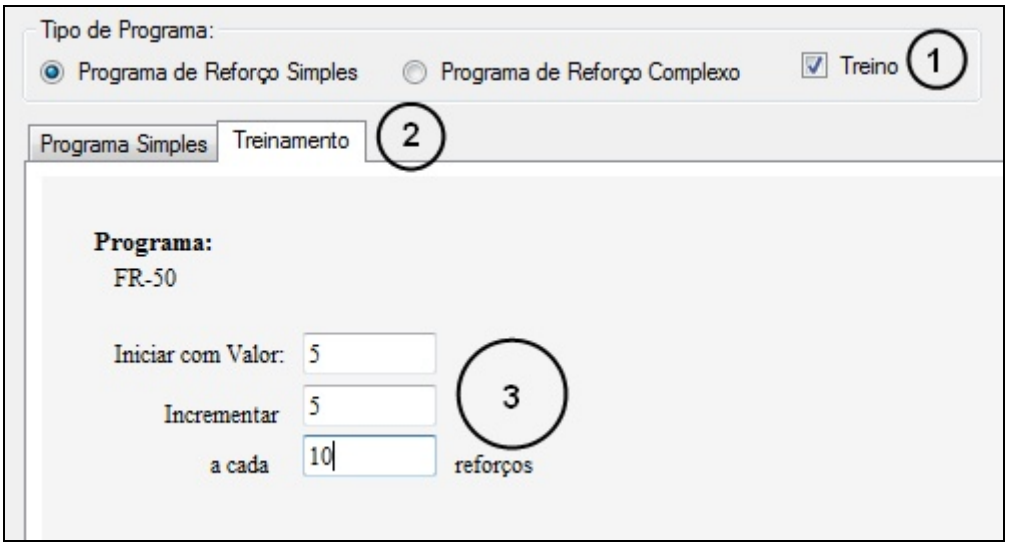
Alguns outros parâmetros do controle experimental em programas complexos podem ser configurados nesse formulário. A opção "*Carry Over*" (Marcador 6 da Figura 4.4), controla o "*reset*" a cada troca de componente. Ao habilitar o *carry over* o valor do número de respostas emitidas (em programas de razão) ou o intervalo de tempo (em programas de intervalo) serão "conduzidos" para o início da próxima apresentação desse mesmo componente. Por exemplo, em um múltiplo FR 60-DRL 20 s um participante ganha pontos no FR (Componente 1) a cada 60 respostas emitidas. Se o participante emite 40 respostas, o intervalo de tempo de exposição àquele componente termina e o *carry over* não está habilitado, quando o componente de FR entrar em vigor novamente o participante teria de emitir 60 respostas para ganhar algum ponto; caso o *carry over* esteja habilitado, quando o componente de FR entrar em vigor novamente, o participante teria de emitir 20 respostas para ganhar algum ponto (i.e., as 40 respostas emitidas antes da mudança do componente de FR – que foram conduzidas para o início dessa nova apresentação do FR – mais as 20 respostas que faltam para completar a razão 60 do FR). A mesma lógica se aplica, neste exemplo, para o intervalo do DRL.

Outro parâmetro é o *time out* (TO) (Marcador 7 da Figura 4.4) – suspensão discriminada das contingências de reforço – que refere-se a um período de tempo sem reforço, durante o qual o organismo, caracteristicamente, não se engaja no comportamento sendo estudado (ou pela extinção durante a apresentação de um estímulo ou pela remoção da oportunidade de responder) e é utilizado como provas (marcadores) em uma série de eventos ou como um método de eliminar os efeitos de um comportamento anterior (Catania, 1998; Ferster & Skinner, 1957). No ProgRef v4, se o TO estiver programado ele entrará em vigor, durante uma sessão experimental, sempre que houver troca entre diferentes componentes (i.e., se houver mudança do Componente 1 para o 2 ou vice-versa; ele não vigorará se um mesmo componente for repetido mais de uma vez). Quando o TO entrar em vigor, a tela do monitor ficará preta e apenas a palavra "AGUARDE" aparecerá escrita em vermelho no centro da tela

(remoção da oportunidade de responder). O período de TO não é computado para o cálculo da duração da sessão.

O problema de pesquisa a ser investigado por um pesquisador, tanto em programas de reforço simples quanto nos complexos, pode exigir um valor considerado alto como parâmetro dos programas FR, FI, FT, DRL ou DRH. Para evitar que o comportamento do participante entre em extinção antes que o comportamento faça contato com a contingência programada, o ProgRef v4 possibilita o "treino" desses programas, no qual o parâmetro da contingência de reforço é incrementado gradualmente. A opção "Treino" fica disponível na parte superior do formulário de configuração da sessão experimental (Marcador 1 da Figura 4.5). Ao ser habilitado uma nova aba "Treinamento" irá aparecer. A Figura 4.5, exibe um exemplo de treinamento de um programa simples FR 50 (Marcador 2 da Figura 4.5). Os critérios de treino são configurados através de uma regra lógica observada na Figura 4.5 (Marcador 3), os parâmetros são: (1) valor inicial do programa de reforço; (2) O valor a ser incrementado; (3) critério para o incremento (quantidade de reforços). Neste exemplo a sessão experimental iniciaria com um FR 5 e, a cada 10 reforços recebidos, a razão aumentaria em 5 unidades até que uma razão 50 fosse atingida. Caso o experimentador, neste mesmo exemplo, tivesse escolhido incrementar a razão em 7 unidades, a lógica seria a mesma, com a diferença que a razão passaria pelos valores 5-12-19-26-33-40-47 e, no último incremento, apenas três unidades seriam acrescentadas, atingindo o valor 50.

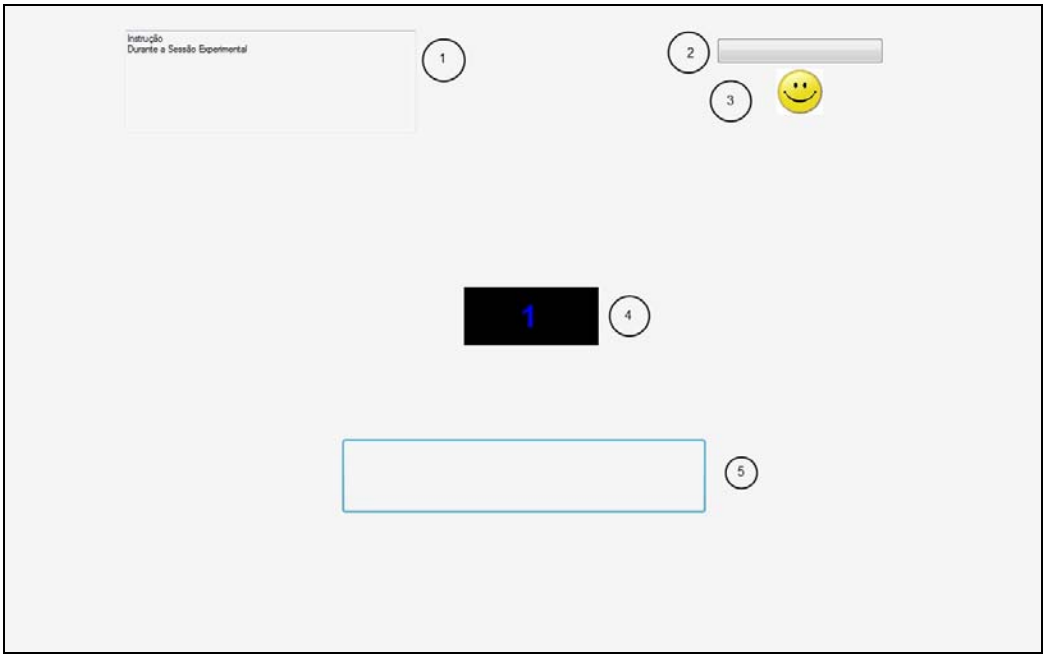
**Figura 4.5** - Programação do Treinamento.



*Sessão Experimental*

A tela da sessão experimental é apresentada na Figura 4.6. Ela é bastante semelhante à tela da sessão experimental do ProgRef v3.

**Figura 4.6** - Tela da Sessão Experimental




Na parte superior esquerda (Marcador 1 da Figura 4.6) haverá uma janela de instrução que ficará disponível para o participante durante toda a sessão – caso ela tenha sido programada (Marcador 19 da Figura 4.2). Em um programa complexo, cada componente poderá ter instruções distintas. Na parte superior direita da Figura 4.6, será apresentado o botão da resposta de consumação (Marcador 2 da Figura 4.6) e, abaixo dele, será exibido o *smile* – quando a exigência do programa de reforço em vigor for cumprida (Marcador 3 da Figura 4.6). Se o *smile* estiver na tela, uma pressão no botão de resposta de consumação fará o *smile* desaparecer e pontos serem creditados no contador (Marcador 4 da Figura 4.6). Caso o *smile* não esteja presente, nehuma consequência está programada para pressões no botão de respostas de consumação mas, diferentemente da versão anterior do *software*, elas serão registradas. O botão de respostas (*operandum*) (Marcador 5 da Figura 4.6) é exibido na parte inferior. Todas as respostas emitidas nesse botão são registradas.

Ao término da sessão experimental é exibida uma mensagem de agradecimento ao participante com a instrução para que chame o experimentador, além de mostrar os pontos obtidos durante a sessão.

*Considerações Finais do Prog Ref v4*

As informações apresentadas anteriormente servem como um guia para programação e configuração de arranjos experimentais e teve por objetivo apresentar as principais funcionalidades e possibilidades do *software*. Os principais formulários foram apresentados e discutidos e, com essas informações é possível para um usuário programar e utilizar todos os recursos do programa.

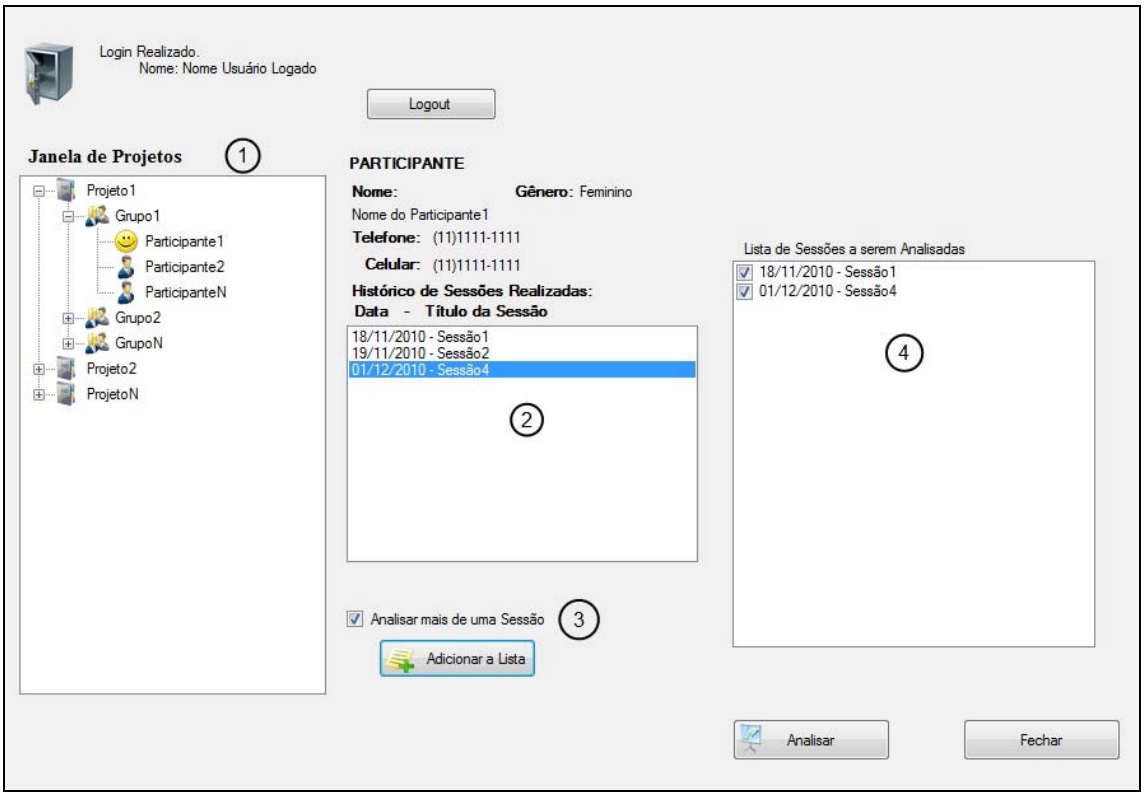
Os dados coletados durante a sessão experimental são salvos em um banco de dados. Caso ocorra algum imprevisto (e.g., queda de energia, comportamento inesperado do sistema operacional etc.) os dados coletados até o momento não são perdidos. Na próxima vez que o ProgRefv4 for executado os dados serão salvos e uma mensagem informando do ocorrido será apresentada para o usuário. Outra opção de cancelar uma sessão em andamento é através da tecla de atalho (ALT+Q), neste caso os dados são salvos no mesmo instante.

## *Análise de Dados (ProgRef_DA v1)*

O programa ProgRef_DA v1 possibilita que os dados coletados no ProgRef v4 sejam visualizados em gráfico cumulativo de repostas e em tabelas. A tela inicial do programa ProgRef_DA v1 mantem o mesmo padrão de organização do ProgRef v4. A Figura 4.7 exibe a tela de abertura após realizado o *login.* A “Janela de Projetos” (Marcador 1 da Figura 4.7) é a mesma descrita no ProgRefv4. Ao selecionar um participante, são apresentadas todas as sessões realizadas na lista “Histórico de Sessões Realizadas” (Marcador 2 da Figura 4.7).

O tratamento dos dados pode ser realizado por sessão ou selecionando várias delas. Para analisar apenas uma sessão, basta selecioná-la e clicar no botão [Analisar]. Se desejar analisar diversas sessões experimentais, deve-se habilitar a opção “Analisar mais de uma sessão experimental” (Marcador 3 da Figura 4.7). Essa ação torna o botão [Adicionar a Lista] (Marcador 3 da Figura 4.7) e a “Lista de sessões a serem analisadas” (Marcador 4 da Figura 4.7) visíveis. O botão [Adicionar a Lista] transfere as sessões da lista de histórico para a lista de sessões a serem analisadas, caso o experimentador não queira mais fazer o tratamento de dados de uma determinada sessão basta desabilitar a opção ao lado do título de cada sessão.

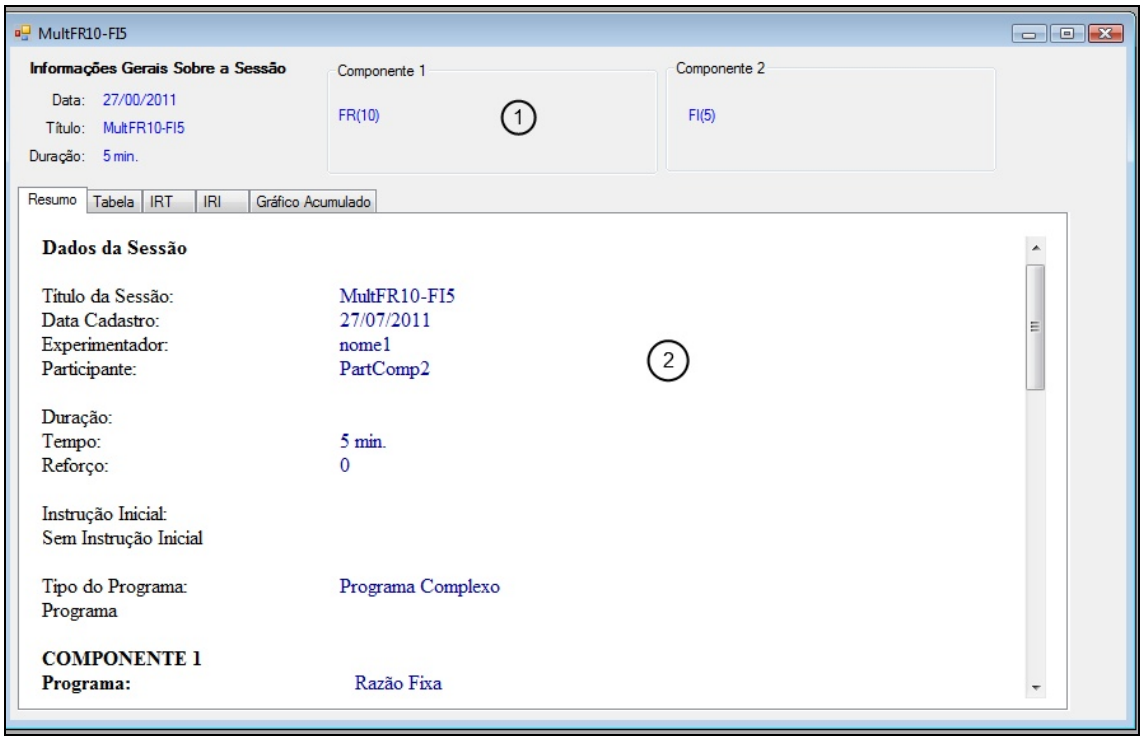
**Figura 4.7** - Tela Inicial do Programa ProgRef_DA v1




A janela de análise será aberta contendo uma ou mais sessões, dependendo da configuração inicial. No caso de mais sessões serão abertas várias janelas como a da Figura 4.8. Na parte superior são exibidas as informações gerais da sessão, com objetivo de ser um acesso rápido para orientar e facilitar a leitura das tabelas e do gráfico (Marcador 1 Figura 4.8). As páginas facilitam a navegação entre as possíveis formas de tratamento dos dados: (1) Resumo; (2) Tabela; (3) IRT; (4) IRI e (5) Gráfico Acumulado.

A primeira página “Resumo” apresenta todos os parâmetros configurados da sessão experimental (Marcador 2 da Figura 4.8).

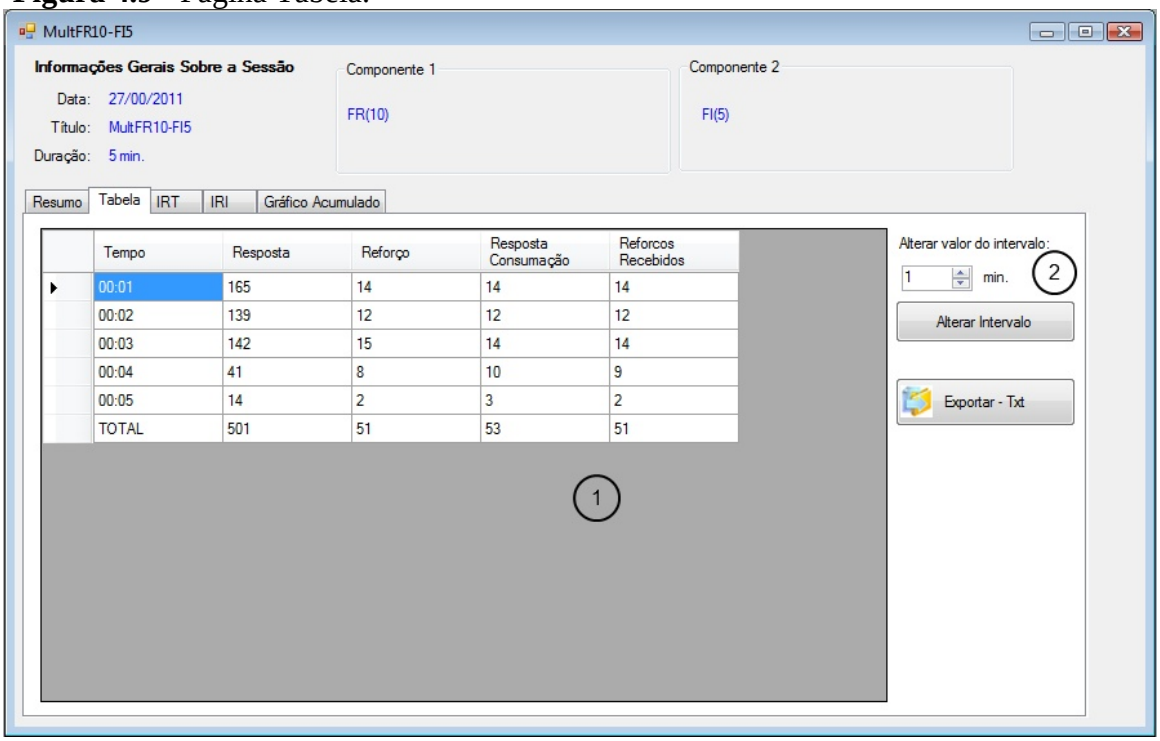
**Figura 4.8** - Tela do resumo da sessão.




Na página “Tabela”, exibida na Figura 4.9, são apresentadas em colunas a unidade de tempo selecionada para análise (em minutos) (Marcador 2 da Figura 4.9), o número de respostas emitidas na unidade de tempo selecionada, o número de reforços obtidos na sessão, número de vezes que a resposta de consumação foi emitida (mesmo que não houvesse o *smile* disponível) e o número de reforços recebidos (i.e., para o qual houve resposta de consumação). Por exemplo, na Figura 4.9, no terceiro minuto da sessão o participante teria obtido 15 smiles (ver a terceira coluna denominada “Reforço”) mas “consumiu”(i.e., pressionou o botão de resposta de consumação após o aparecimento do smile) 14 deles (ver a quinta coluna denominada “Reforços Recebidos”). No minuto seguinte, o participante “consumiu” este reforço e ganhou (e consumiu) outros oito reforços – por isso aparece o número 8 na coluna “Reforço” e 9 na coluna “Reforços Recebidos”.

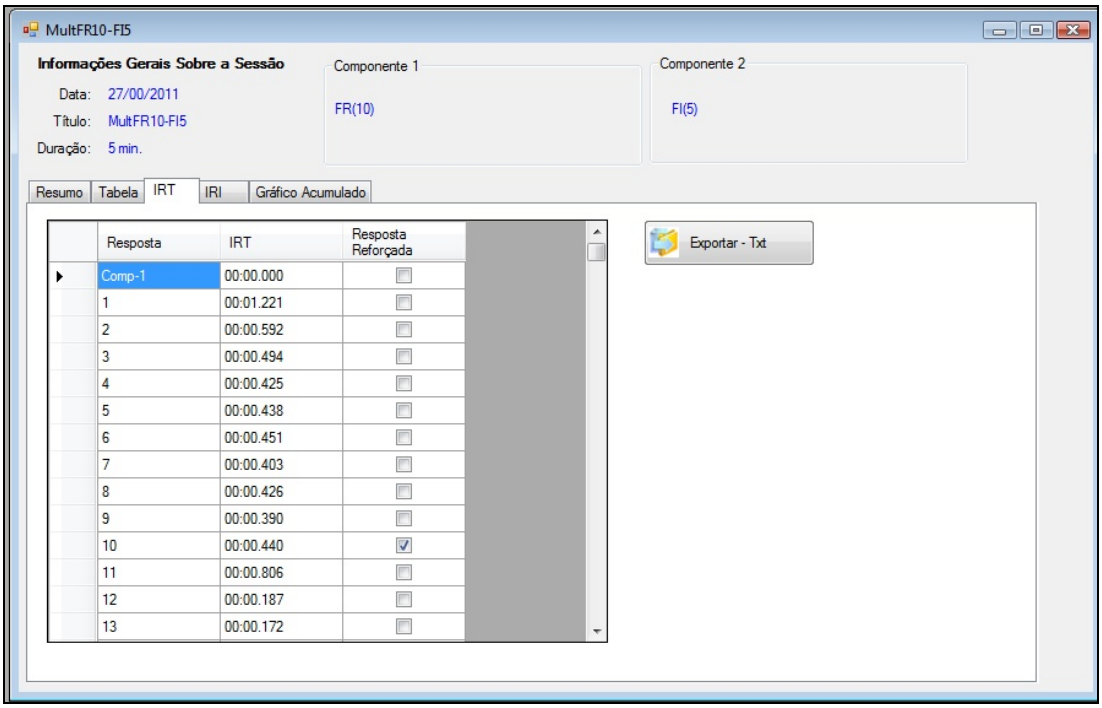
O botão [Exportar – Txt] permite que o conteúdo da tabela (Marcador 1 da Figura 4.9) seja transferido para um arquivo texto com separadores entre colunas configuráveis. Todas as tabelas podem ser exportadas para formato txt, possibilitando que análises que atendam ao interesse do pesquisador seja realizada por outro *software* e outra lógica (e.g. Microsoft Excel®).

**Figura 4.9** - Página Tabela.




Também é possível gerar uma tabela com os intervalos entre respostas (Figura 4.10). Na primeira coluna “Respostas” é apresentada a contagem das respostas, na segunda coluna “IRT” é apresentado o tempo entre uma resposta e outra, e na terceira coluna “Resposta Reforçada” indica se aquela resposta tornou disponível ou não o reforço. Quando o experimentador coleta dados utilizando um programa Tand é exibida uma quarta coluna, indicando em qual resposta cumpriu a contingencia do primeiro ELO do Tand.

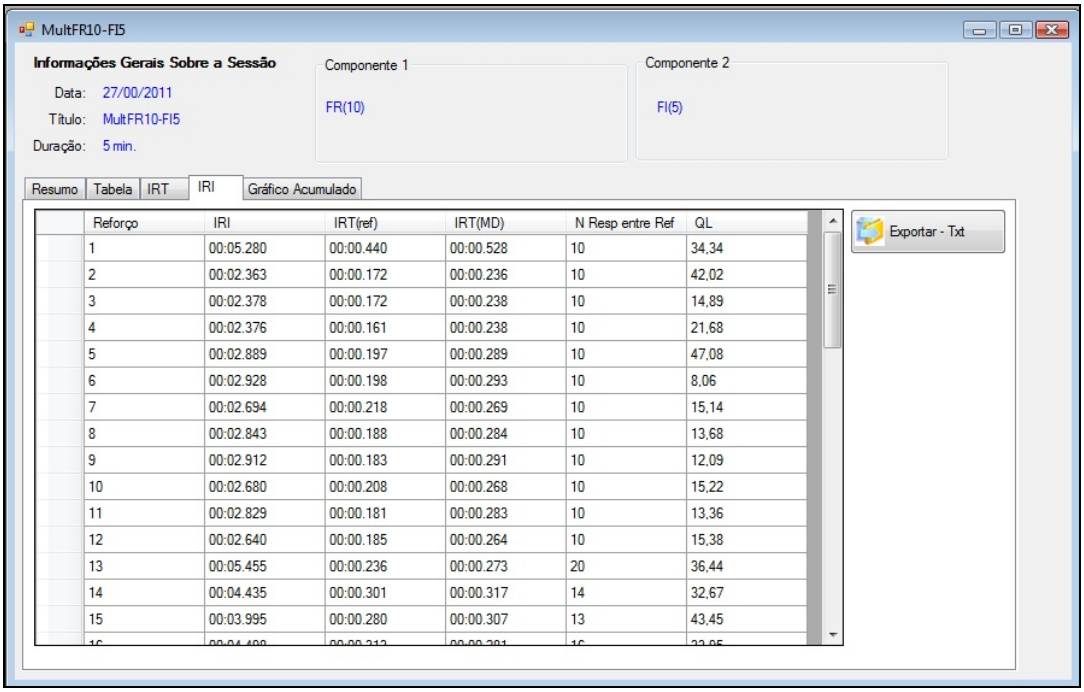
**Figura 4.10** - Página IRT.




Na Figura 4.10 observa-se a inscrição “Comp-1” na primeira linha e primeira coluna, essa marcação ocorre quando os dados a serem tratados forem de um programa complexo. Assim, na tabela IRT toda troca de componente é sinalizada.

O intervalo entre reforços pode ser visualizado na página IRI (Figura 11). A primeira coluna “Reforço” apresenta a contagem dos reforços, na segunda coluna “IRI” calcula o tempo entre os reforços, a terceira coluna “IRT(ref)” indica o valor do IRT da resposta que liberou o reforço, a quarta coluna “IRT(MD)” apresenta o valor do IRT médio (média aritmética) dos IRT’s entre os reforços, a quinta coluna “N° Resp entre Ref” quantas respostas foram emitidas entre os reforços e a sexta coluna “QL” *quarter life* .

**Figura 4.11** - Página IRI

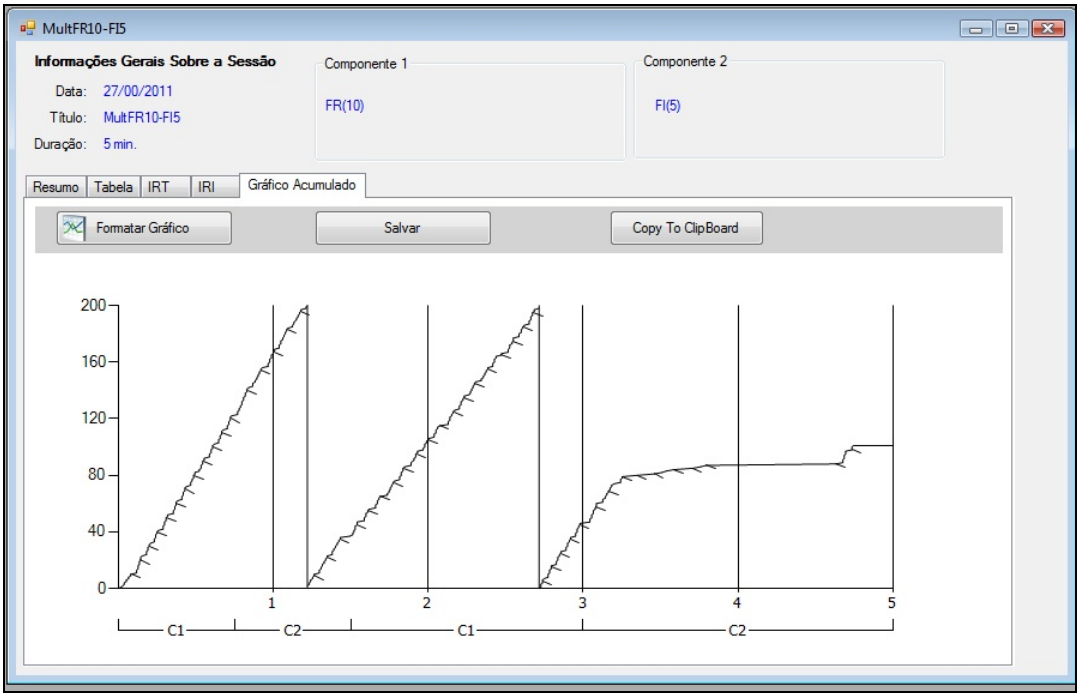



| Reforço | IRI | IRT(ref) | IRT(MD) | N Resp entre Ref | QL |
|---|---|---|---|---|---|
| 1 | 00:05.280 | 00:00.440 | 00:00.528 | 10 | 34,34 |
| 2 | 00:02.363 | 00:00.172 | 00:00.236 | 10 | 42,02 |
| 3 | 00:02.378 | 00:00.172 | 00:00.238 | 10 | 14,89 |
| 4 | 00:02.376 | 00:00.161 | 00:00.238 | 10 | 21,68 |
| 5 | 00:02.889 | 00:00.197 | 00:00.289 | 10 | 47,08 |
| 6 | 00:02.928 | 00:00.198 | 00:00.293 | 10 | 8,06 |
| 7 | 00:02.694 | 00:00.218 | 00:00.269 | 10 | 15,14 |
| 8 | 00:02.843 | 00:00.188 | 00:00.284 | 10 | 13,68 |
| 9 | 00:02.912 | 00:00.183 | 00:00.291 | 10 | 12,09 |
| 10 | 00:02.680 | 00:00.208 | 00:00.268 | 10 | 15,22 |
| 11 | 00:02.829 | 00:00.181 | 00:00.283 | 10 | 13,36 |
| 12 | 00:02.640 | 00:00.185 | 00:00.264 | 10 | 15,38 |
| 13 | 00:05.455 | 00:00.236 | 00:00.273 | 20 | 36,44 |
| 14 | 00:04.435 | 00:00.301 | 00:00.317 | 14 | 32,67 |
| 15 | 00:03.995 | 00:00.280 | 00:00.307 | 13 | 43,45 |

O *quarter life* é um índice para análise de desempenho do responder em FI, no entanto o programa de análise calcula para todos os programas. Ele representa de forma quantitativa como foi a distribuição das respostas durante um intervalo entre reforços.

Outra página de análise é o gráfico de registro cumulativo “Gráfico Acumulado” das respostas Figura 4.12. O gráfico é construído para representar as respostas momento a momento. Facilitando a visualização de possíveis mudanças no comportamento em estudo. As indicações C1 e C2 (lê-se componente 1 e componente 2) representam as trocas dos componentes em um programa complexo.

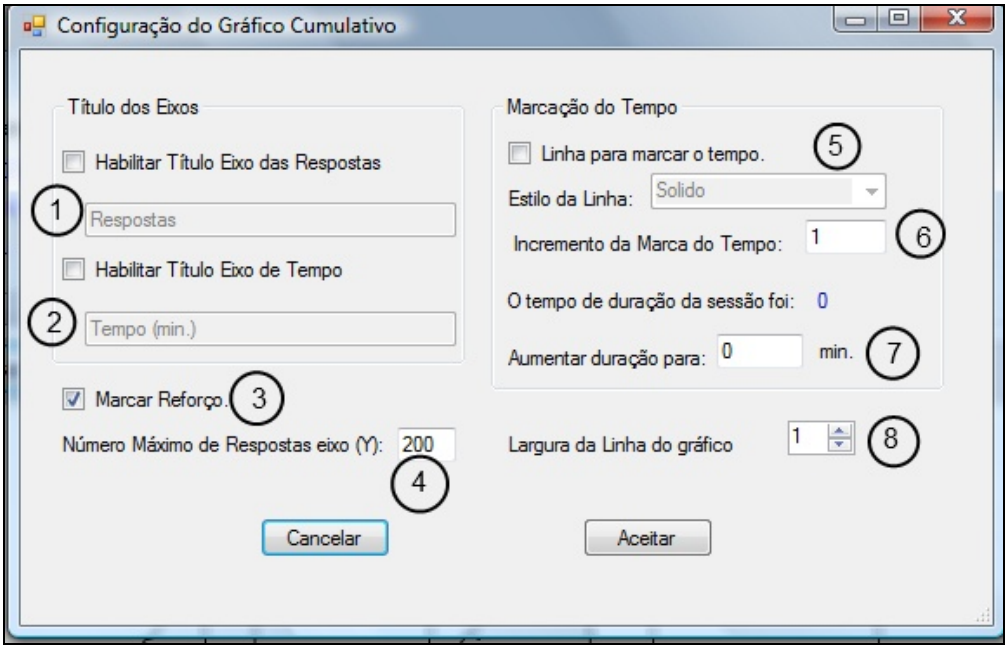
**Figura 4.12** - Página do Gráfico – Registro cumulativo das respostas.




A Figura 4.13 exibe o formulário de configuração de alguns parâmetros editáveis do gráfico. No eixo das ordenadas encontra é apresentado o número de respostas, o campo “Habilitar Título Eixo de Resposta” (Marcador 1 da Figura 4.13) possibilita que o usuário insira um título a esse eixo. O valor máximo de respostas que controla a volta do gráfico a zero pode ser configurado pelo campo “Número máximo de respostas eixo (y)” (Marcador 4 da Figura 4.13). No eixo das abscissas é apresentado o tempo em minutos pode ser inserido um “Habilitar Título Eixo de Tempo” (Marcador 2 da Figura 4.13). Note-se, no exemplo a marcação de tempo esta de minuto a minuto e a sessão tem duração de 5 minutos. Tanto o estilo da linha que pode assumir os valores: Nenhum; Sólido; Tracejado; Traço ponto; Traço Ponto Ponto e Pontilhado. Quanto ao intervalo das marcas de tempo (Marcador 6 da Figura 4.13) pode ser configurada a unidade desejada, sempre em minutos (e.g., pode-se ter linha tracejada de 5 em 5 minutos).

Também é possível aumentar a duração da sessão, por exemplo essa sessão acabou em 5 minutos, pode-se configurar para o término 10 minutos, nesse caso seria traçado uma linha reta até os 10 minutos. Essa configuração facilita comparações entre sessões com diferentes durações.

**Figura 4.13** - Formulário de configuração do gráfico.




A largura da linha do gráfico também pode ser aumentada, facilitando a visualização do gráfico quando for impresso. Também é possível definir se deseja ou não visualizar as marcas de reforço no gráfico.

O programa ProgRef_DA v1 procurou atender as principais exigências dos experimentadores na análise e permitir a exportação dos dados para que outros tratamentos dos dados sejam realizadas.

## CONSIDERAÇÕES FINAIS

Os programas de reforço se mostram uteis na investigação do comportamento. As simplificações proporcionadas por esse modelo de estudo auxiliam no controle e predição do comportamento. Os arranjos experimentais podem assumir as mais diversas configurações. O avanço tecnológico, tanto em termos abstratos (e.g., modelos preditivos, aplicações) quanto equipamentos (e.g., caixa de Skinner, *softwares*) auxiliam no controle das sessões experimentais. Essas tecnologias otimizam, possibilitam uma replicação facilitada, e também validam, propicia controle para que as variáveis em estudo esteja devidamente isolada de interferências.

A informática ou, mais especificamente no contexto desse trabalho, o desenvolvimento de *softwares*, tem prestado grande auxílio para a realização de pesquisas científicas, acelerando ou possibilitando o estudo de determinado fenômeno.

A utilização de *softwares* pela Análise Experimental do Comportamento tem-se mostrado benéfica. Muitas pesquisas utilizam *softwares* específicos para a coleta e tratamento dos dados. A produção científica apresentada pelo *software* ProgRef v3 exemplifica esses ganhos e justifica o investimento para o desenvolvimento de uma nova versão.

O ProgRefv4 foi desenvolvido para garantir tanto a demanda por *softwares* para pesquisas com programas de reforço com grande liberdade de configuração dos arranjos experimentais, além de conseguir unir a característica de ser de fácil manuseio.

Além do *software* propriamente dito, as discussões sobre o uso das tecnologias que a Analise Experimental do Comportamento, em específico, e a Psicologia, de um modo geral, veem apresentando se mostram oportunas.

# REFERÊNCIAS


Alloway, T., Wilson G., Graham, J., & Krames, L. (2000). *Sniffy: The Virtual Rat - Pro Version*. Belmont, CA: Wadsworth/ Thomson Learning.

Anderson, M.D. & Hornby, P.A. (1996). Computer attitudes and the use of computers in psychology courses. *Behavior Research methods, instruments & Computers*, 28(2), 341-346.

Ator, N. A. (1991). Subjects and instrumentation. In I. H. Iversen & K. A. Lattal (Eds.), *Experimental Analysis of Behavior, Part 1* (pp. 1-62). New York, NY: Elsevier Science.

Azzi, R., Rocha & Silva, M. I., Bori, C. M., Fix, D. S. R., & Keller, F. S. (1963). Suggested Portuguese translations of expressions in operant conditioning. *Journal of the Experimental Analysis of Behavior*, 6(1), 91-94.

Beagley, W. K. (2001). Why we need more psychology programmers/ El Knife, a data utility for transforming spreadsheets. *Behavior Research methods, instruments & Computers*, 33(2), 97-101.

Becker, R. M.; Xavier, J. M.; Soares, P. G. ; Banaco, R. A. ; Costa, C. E. (2006). Efeito do custo da resposta sobre o responder de humanos em FI, após uma história de responder em FR, quando pontos foram trocados por dinheiro. *Anais do XV EAIC e VI EPUEPG*, Ponta Grossa, PR.

Braden, D. (1996). *Ser cientista: O espírito de aventura em ciência e tecnologia* (Monica Saddy, Trad.). Campinas: Papirus. (original publicado em 1994).

Campbell-Kelly, M. (2009). A Origem da Computação. *Scientific American Brasil*, 89, 48-55.

Camara, F. (2006). *Orientação a Objeto com .NET* (2ª ed.). Florianopolis, SC: Visual Books.

Catania, A. C. (1998). *Learning* (4ª ed.). New Jersey: Prentice Hall.

Catania, A. C. & Reynolds, G. S. (1968). A quantitative analysis of the responding maintained by interval schedules of reinforcement. *Journal of the Experimental Analysis of Behavior*, 11(3), 327-383.

Chance, P. (2009). *Learning and Behavior: Active Learning Edition* (6ª ed.). Belmont: Wadsworth. Disponível em: http://books.google.com.br/books?id=SWy3uoXJw_4C&printsecfrontcover&dq=%22Learning+and+Behavior%22&source=bl&ots=4ltW42W_WT&sig=_UdmNU6RZ9utIF4bFB7K9vhMlvY&hl=pt-BR&ei=BPn3S6WNFMH7lwf_wf3iCg&sa=X&oi=book_result&ct=result&resnum=5&ved=0CDEQ6AEwBA#v=onepage&q&f=false. (Acessado em: 15/05/2010).

Churchland, P. M. (2004). *Matéria e Consciência: uma introdução contemporânea à filosofia da mente*. São Paulo: UNESP.

Chiesa, M. (1994). *Radical behaviorism: The philosophy and the science*. Boston, MA: Authors Cooperative.

Collins, Grahan P. (2008). Uma Máquina de Descobertas. *Scientific American Brasil*, 70, 48-51.

Costa, C. E. (2006). *Softwares* para pesquisa: Relato de experiência. In: O. Z. Prado, E. Fortim; L. Consentino (Orgs.), *Psicologia & Informática: Produções do III PsicoInfo e II Jornada do NPPI* (pp. 168-177). São Paulo: CRP/SP.

Costa, C. E. & Banaco, R. A. (2002). ProgRef v3: Sistema Computadorizado para coleta de dados sobre programas de reforço com humanos – recursos básicos. *Revista Brasileira de Terapia Comportamental e Cognitiva*, 4(2), 173-192.

Costa, C. E. & Banaco, R. A. (2003). ProgRef v3: sistema computadorizado para coleta de dados sobre programas de reforço com humanos - recursos adicionais. *Revista Brasileira de Terapia Comportamental e Cognitiva*, 5(2), 219-229.

Costa,C. E.; Patsko, C. H. & Becker, R. M. (2007). Desempenho em FI com Humanos: Efeito da Interação da Resposta de Consumação e do Tipo de Instrução. *Interação em Psicologia*, 11(2), 175-186.

Costa, V. C. I.; Paula, E.; Xavier, G. F. & Bueno, J. L. O. (2007). Programa "DRL" para Controle Experimental de Pesquisa em Julgamento Temporal. *Psicologia: Reflexão & Crítica*, 20(3), 507-512.

Costa, C. E.; Soares, P. G. e Ramos, M. N. (submetido). Controle de estímulos e história comportamental: uma replicação sistemática de Freeman e Lattal (1992). *Temas em Psicologia*.

Dandurand, F.; Shultz, T. R. & Onishi, K. H. (2008). Comparing online and lab methods in a problem-solving experiment. *Behavior Research Methods*, 40(2), 428-434.

Dawkins, R. (2001). *O Relojoeiro Cego: A Teoria da Evolução contra o Desígnio Divino*. São Paulo: Companhia das Letras.

De Freitas, L. A. B. (2009). *O efeito da consequência programada sobre a estabilidade da taxa de respostas em FI*. Dissertação (Mestrado em Análise do Comportamento). Universidade Estadual de Londrina, Londrina.

Debert, P. & Andery, M. A. P. A. (2005). Infravermelho de baixo custo para estabelecer respostas de focinhar em ratos. *Revista Brasileira de Terapia Comportamental e Cognitiva*, 7(2), 263-266.

Delyra, J. L. (1997). A universidade e a revolução informática. *Revista USP*, 35, 77-85.

Donahoe, J. W. & Palmer, D. C. (1994). *Learning and Complex Behavior*. Massachusets: Allyn and Bacon.

Eckerman, D. A. Vasconcelos, L. A. & Gimenes, L. S. (2006). Temos o prazer em trazer CyberRat para a Escola. In: O. Z. Prado, E. Fortim, L. Consentino (Orgs.), *Psicologia & Informática: Produções do III PsicoInfo e II Jornada do NPPI* (pp. 178-185). São Paulo: CRP/SP.

Fernandes, L. (2003). Abacus: The Art of Calculating with Beads. Disponível em: http://www.ee.ryerson.ca:8080/~elf/abacus/index.html. (Acessado em: 15/11/2009).

Ferster, C. B., Culbertson, S., & Boren, M. C. P. (1979). *Princípios do Comportamento* (M. I. R. Silva, M. A. C. Rodrigues & M. B. L. Pardo, Trans.). São Paulo, SP: Hucitec. (original publicado em 1968).

Ferster, C. B. & Skinner, B. F. (1957). *Schedules of Reinforcement*. New York: Appleton Century Crofts.

Fleshler, M. & Hoffman, H. S. (1962). A progression for generating variable-interval schedules. *Journal of the Experimental Analysis of Behavior*, 5(4), 529-530.

Freeman, T. J., & Lattal, K. A. (1992). Stimulus control of behavioral history. *Journal of the Experimental Analysis of Behavior*, 57(1), 5-15.

Flynn, I. M. & McHoes, A. M. (2002). *Introdução aos Sistemas Operacionais*. São Paulo, SP: Thompson.

Galvez, J. A. (2007). *Dicionário Larousse inglês-português, português-inglês avançado*. (3ª reimpressão). São Paulo: Larousse.

Garofalo, G. L. (2004) Considerações sobre Microeconomia. In: D. B. Pinho & M. A. S. Vasconcellos (Org.), *Manual de Economia. Equipe de Professores da USP*. São Paulo, SP: Saraiva.

Goyos, C. & Almeida, J. C. B. (1994). Mestre (Version 1.0) [*computer software*]. São Carlos, Brasil: Mestre *Software*.

Goyos, C. (2004). Mestre: Um recurso derivado da interface da Análise Comportamental com a informática para aplicações educacionais. In: M. M. C. Hübner & M. Marinotti (Orgs.), *Análise do Comportamento para Educação: Contribuições Recentes* (pp. 285-305). Santo André, SP: ESETEC.

Gleick, J. (1989). *Caos: a criação de uma nova ciência*. Campus. Rio de Janeiro: Campus.

Gollub, L. R. (1991) The use of computers on the control and recording of behavior. In: I. H. Iversen & K. A. Lattal (Ed.), *Experimental Analysis of Behavior* – Part 2 (pp. 155-192). Amsterdam: Elsevier.

Gugik, G.(2009). *A história dos sistemas operacionais*. Disponível em: http://www.baixaki.com.br/info/2031-a-historia-dos-sistemas-operacionais.htm. (Acessado em: 20/02/2010).

IBGE (2007). *Pesquisa Nacional por Amostra de domicilios: Acesso à internet e posse de Telefone Móvel Celular para uso pessoal 2005. Rio de Janeiro, RJ*. Disponível em: www.ibge.gov.br/home/estatistica/populacao/acessoainternet. (Acessado em: 27/01/2010)

Jansen, R. G.; Wiertz, L. F.; Meyer, E. S. & Noldus, L. P. J. J. (2003). Reliability analysis of observational data: Problem, solutions, and *software* implementation. *Behavior Research methods, instruments & Computers*. 35(3), 391-399.

Killeen, P. R. (1985) Reflections on a Cumulative Record. *The Behavior Analyst*, 8(2), 177-183.

Kowaltowski, T. (1996). *John von Neumann: Suas Contribuições à Computação*. Disponível em: www.ic.unicamp.br/~tomasz/projects/vonneumann/artigo.html. (Acessado em: 01/09/2009).

Lattal, K. A. (1991). Scheduling positive reinforcers. In I. H. Iversen & K. A. Lattal (Eds.), *Experimental Analysis of Behavior* (Part 1, pp. 87-134). New York, NY: Elsevier Science.

Lattal, K. A. (2004). Steps and Pips in the history of cumulative recorder. *Journal of the Experimental Analysis of Behavior, 82*(3), 329-355.

Lattal, K. A. (2005). Ciência, tecnologia e Análise do Comportamento. In: J. Abreu-Rodrigues e M. R. Ribeiro (Ed.). *Análise do Comportamento: Pesquisa, Teoria e Aplicação* (pp. 15-26). Porto Alegre, RS: Artmed.

Lattal, K. A. (2006). O lado humano do comportamento animal. *Revista Brasileira de Análise do Comportamento*, 2(1), 1-19.

Lattal, K. A.(2008). JEAB at 50: Coevolution of research and technology. *Journal of the Experimental Analysis of Behavior*, 89(1), 129-135.

Laudares, F.; Lopes, M. C. S. M. & Cruz, F. A. O. (2004). Usando sensores magnéticos em um trilho de ar. *Revista Brasileira de Ensino de Física*, 26(3), p. 233 - 236.

Matthews, B. A., Shimoff, E., Catania, A. C. & Sagvolden, T. (1977). Uninstructed human responding: Sensitivity to ratio and interval contingencies. *Journal of the Experimental Analysis of Behavior*, 27(3), 453-467.

MED Associates. (2003). *MED-PC® for Windows. Published by MED Associates Inc*. [Programmer's Manual] USA: Vermont.

Millenson, J. R. (1975). *Princípios da Análise do Comportamento*. Brasília: Coordenada. (Original publicado em 1967).

Morse, W. H. (1966). *Intermittent Reinforcement*. In W. K. Honig (Ed.), Operant Behavior: Areas of Research and Application (pp. 12-32). New Jersey: Prentice-Hall.

Nicolelis, M. (2011). *Muito além do nosso eu: a nova neurociência que une cérebros e máquinas – e como ela pode mudar nossas vidas*. São Paulo: Companhia das Letras.

Ottoni, E. (2000). EthoLog 2.2: A tool for the transcription and timing of behavior observation sessions. *Behavior Research methods, instruments & Computers*. 32(3), 446-449.

Panetta, P. A. B.; Hora,C. L. & Benvenuti, M. F. L.(2007). Avaliando o papel do comportamento verbal para aquisição de comportamento "supersticioso". *Revista Brasileira de Terapia Comportamental e Cognitiva*, 9(2), 277-287.

Pessoa, C. V. B. & Buffara, A. C. L. (2005). Construção de intervalos variáveis de reforçamento em planilha eletrônica de cálculo. *Revista Brasileira de Terapia Comportamental e Cognitiva*, 7(1), 133-136.

Pimentel, J. R.; Zumpano, V. H. & Yaginuma, L. T. (1989). Trilho de Ar - Uma proposta de baixo custo. *Revista Brasileira de Ensino de Física*, 11(1),15-23.

Pierce, W. D. & Cheney, C. D. (2004). *Behavior Analysis and Learning*. 3ª ed. New Jersey: Lawrence Erlbaum.

Rachlin, H. & Green, L. (1972). Commitment, Choice and Self-Control. *Journal of the Experimental Analysis of Behavior*, 17 (1), 15-22.

Raia, C. P., Shillingford, S. W., Miller Jr., H. L., & Baier, P. S. (2000). Interaction of procedural factors in human performance on yoked schedules. *Journal of the Experimental Analysis of Behavior*, 74(3), 265-281.

Reips, U. D. (2002). Standards for Internet-based experimenting. *Experimental Psychology*, 49, 243-256.

Sebesta, R. W. (2003). *Conceitos de Linguagens de Programação* (5ª ed.). São Paulo: Bookman.

Soares, P. G. (2008). *Controle de estímulos e história comportamental em humanos*. Dissertação apresentada ao Programa de Mestrado em Análise do Comportamento da Universidade Estadual de Londrina. 84 p.

Souza Júnior, E. J. & Cirino, S. D. (2004). Esquemas de Reforçamento. In: C. E. Costa; J. C. Luzia & H. H. N. Sant'Anna (Orgs.), *Primeiros Passos em Análise do Comportamento e Cognição* (Vol. 2, pp.31-42). Santo André: Esetec: Editores Associados.

Skinner, B. F. (1984). Selection by consequences. *The Behavior and brain Sciences*, 7, 477-481. (original publicado em 1981)

Shull R. L. & Lawrence, P. S. (1998). Reinforcement: Schedule Performance. In K. A. Lattal & M. Perone (Eds.), *Handbook of Research Methods in human Operant Behavior*. (pp. 95-130). New York, NY: Plenum Press.

Tomanari, G. Y. & Eckerman, D. A. (2003). O rato *Sniffy* vai à escola. *Psicologia: Teoria e Pesquisa*, 19(2), 154-164.

Wanchisen, B. A.; Tatham, T. A. & Hineline, P. N. (1988). Pigeon's choice in situations of diminishing returns: fixed- versus progressive ratio schedules. *Journal of the Experimental Analysis of Behavior*, 50(3), 375-394.

Weiner, H. (1964). Conditioning history and human fixed-interval performance. *Journal of the Experimental Analysis of Behavior*, 7, 383-385.

Weiner, H. (1965). Conditioning history and maladaptative human operant behavior. *Psychological Reports*, 17, 935-942.

Weiner, H. (1969). Controlling human fixed-interval performance. *Journal of the Experimental Analysis of Behavior*, 12, 349-373.

Weiner, H. (1970). Human behavioral persistence. *The Psychological Record*, 20, 445-456.

# APÊNDICE

# APÊNDICE A
FIGURAS DO CAPÍTULO 4

**Figura 4.1** - Tela inicial do ProgRef v4, após o *login* do usuário.

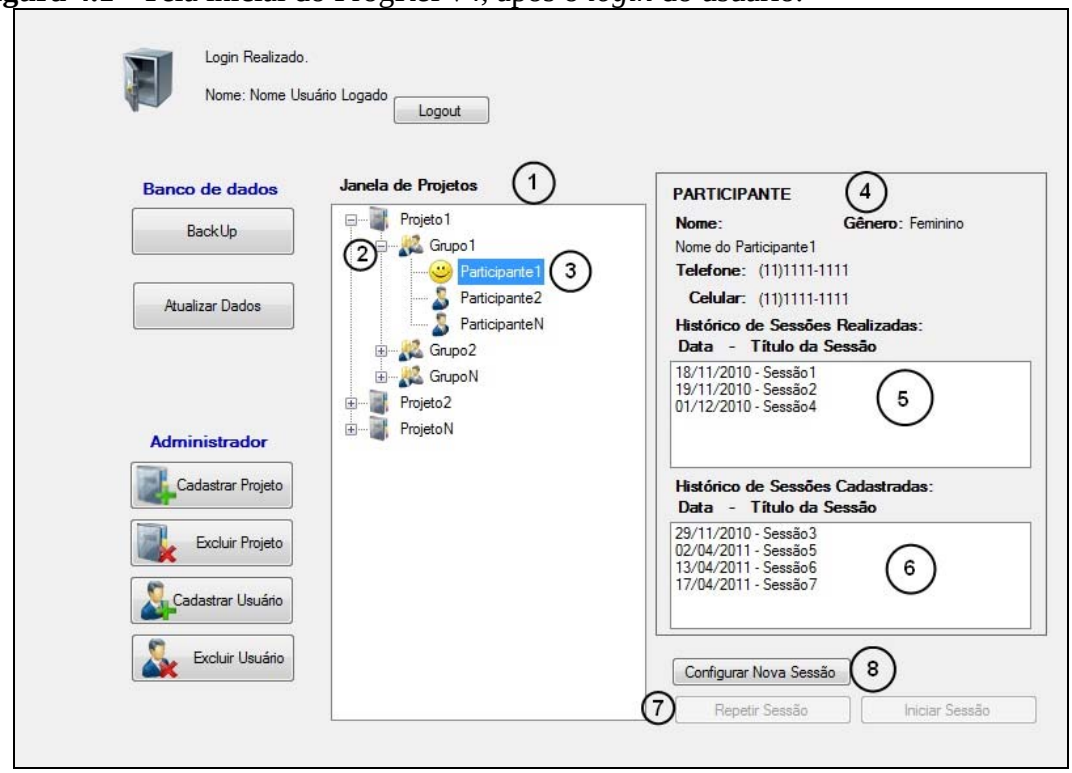



**Figura 4.2** - Formulário de configuração da sessão experimental. Marcadores (ver descrição no texto).

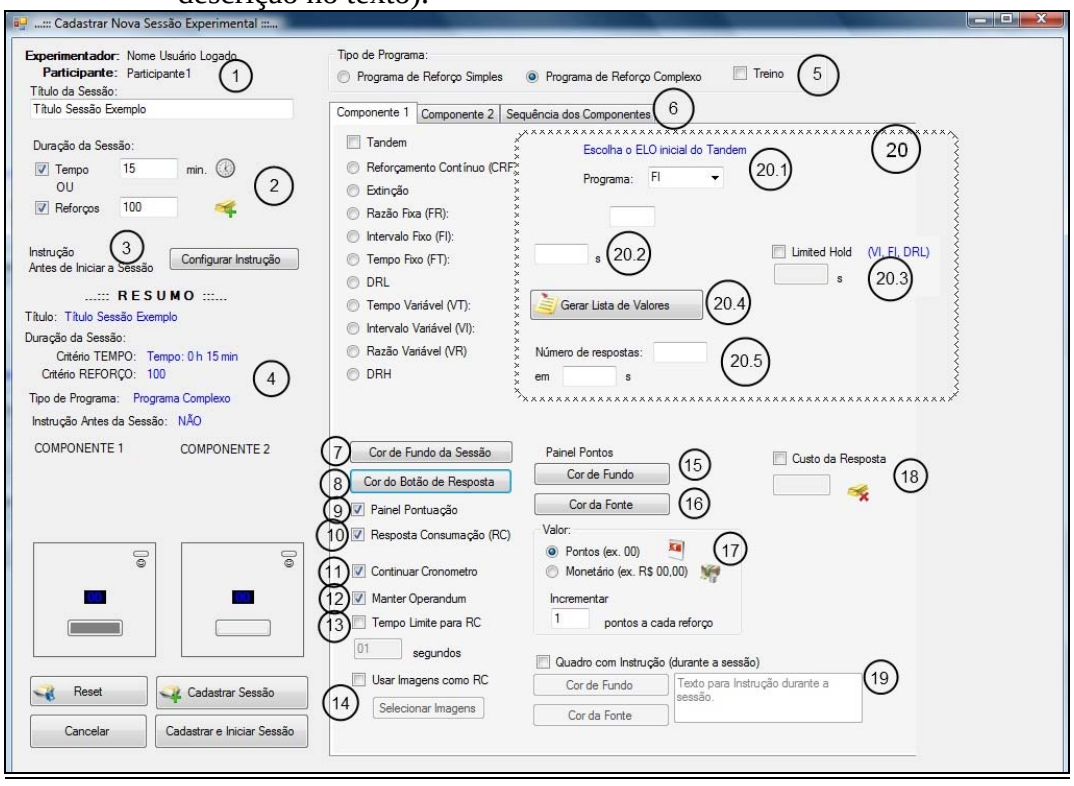


**Figura 4.3** - Formulário Sorteio.

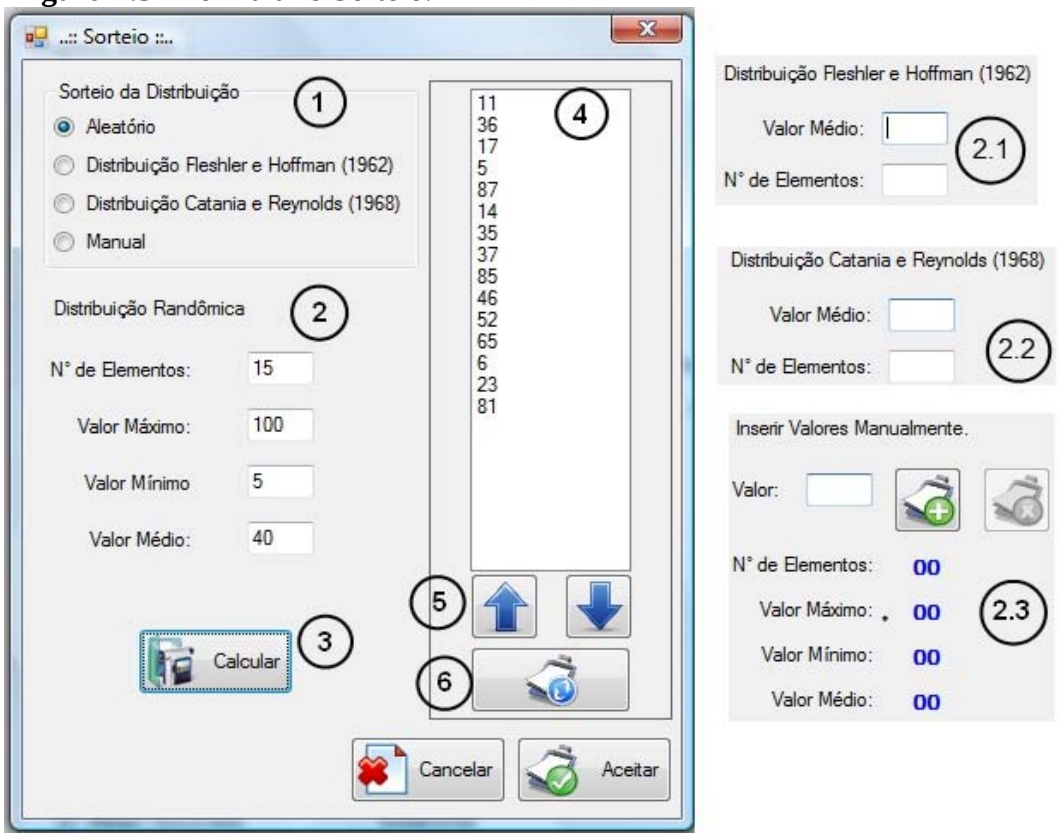



**Figura 4.4** - Configuração da sequencia dos componentes de um programa complexo.

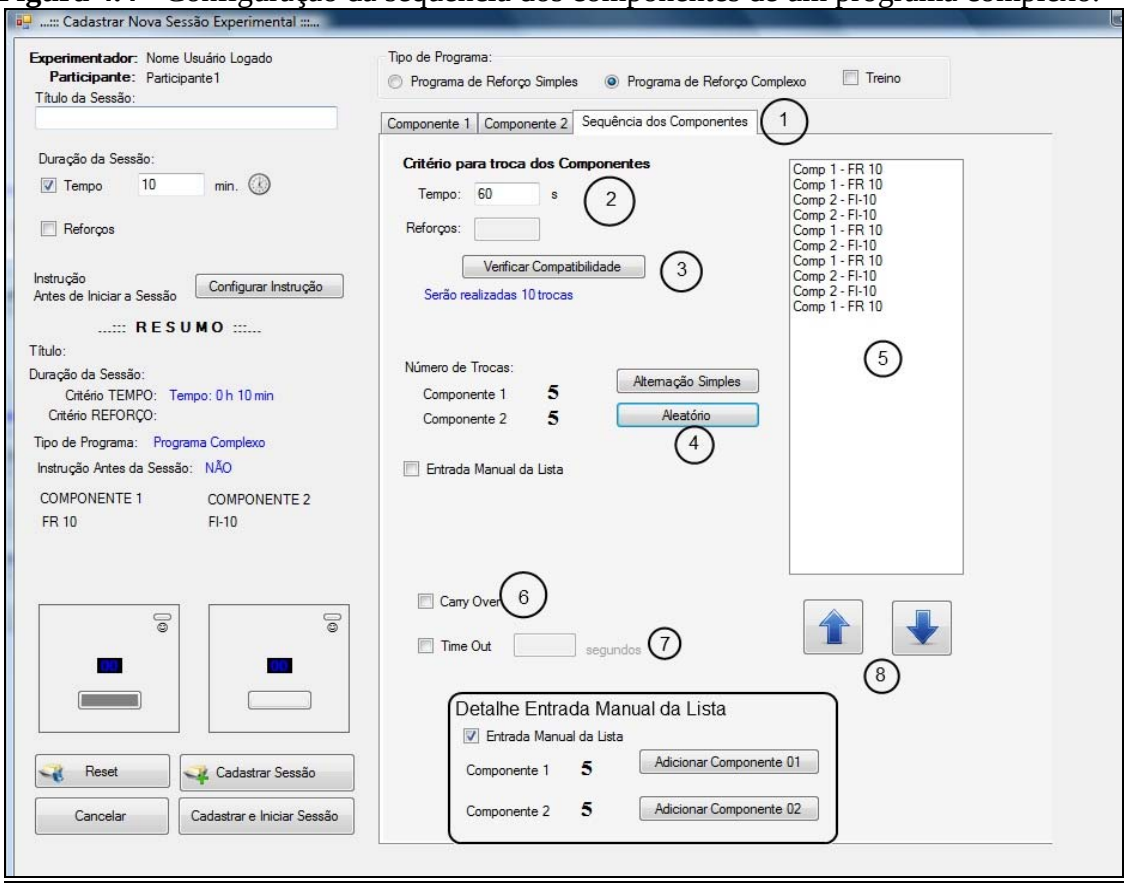


**Figura 4.5** - Programação do Treinamento.

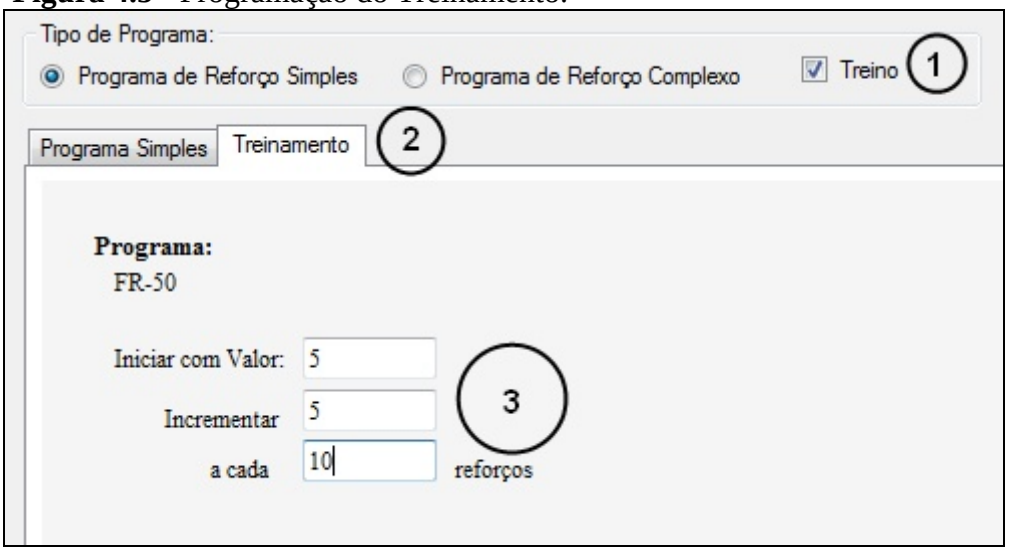



**Figura 4.6** - Tela da Sessão Experimental

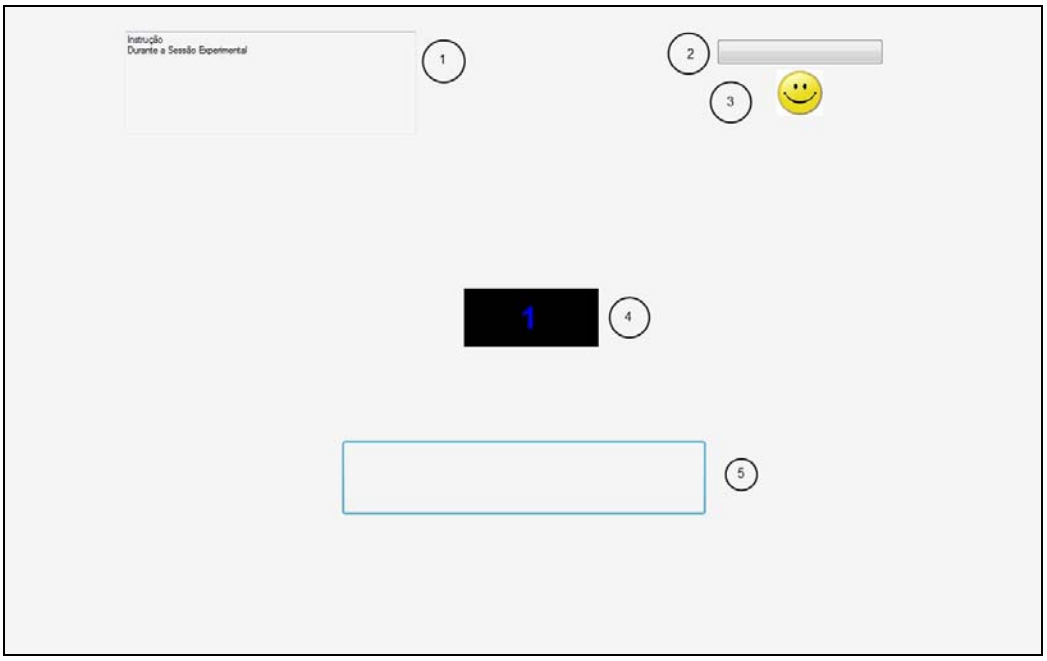


**Figura 4.7** - Tela Inicial do Programa ProgRef_DA v1

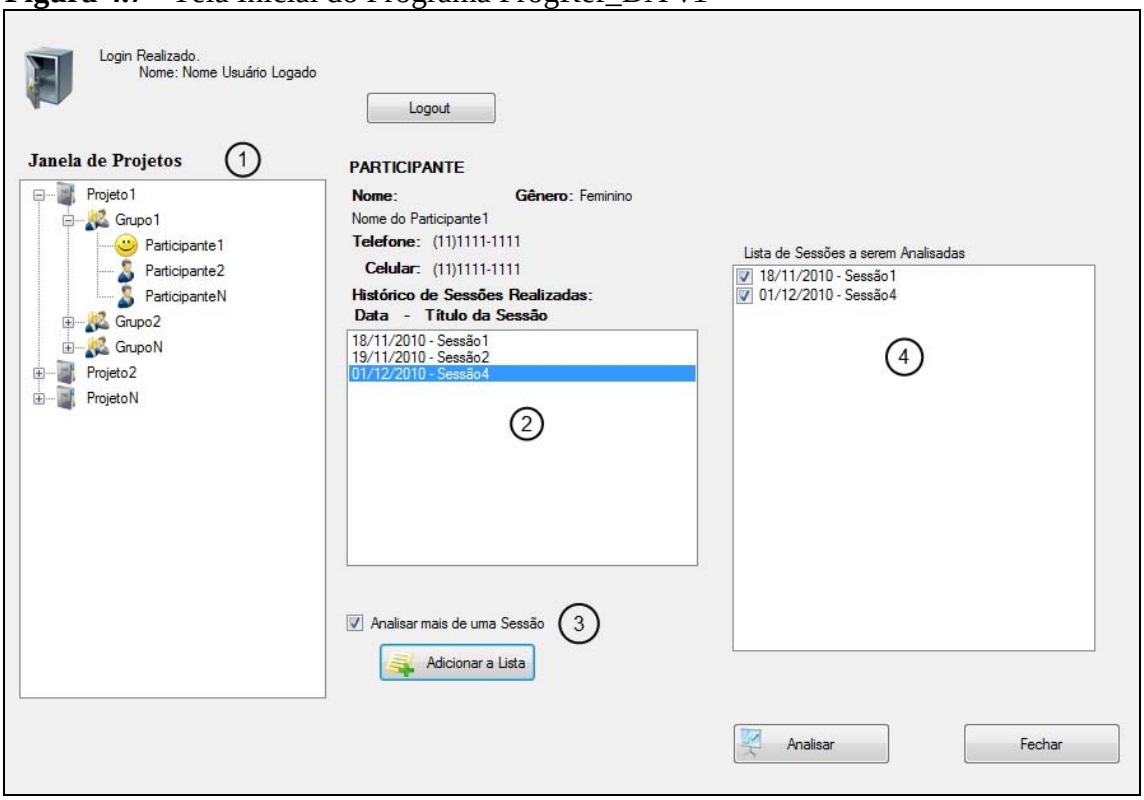



**Figura 4.8** - Tela do resumo da sessão.

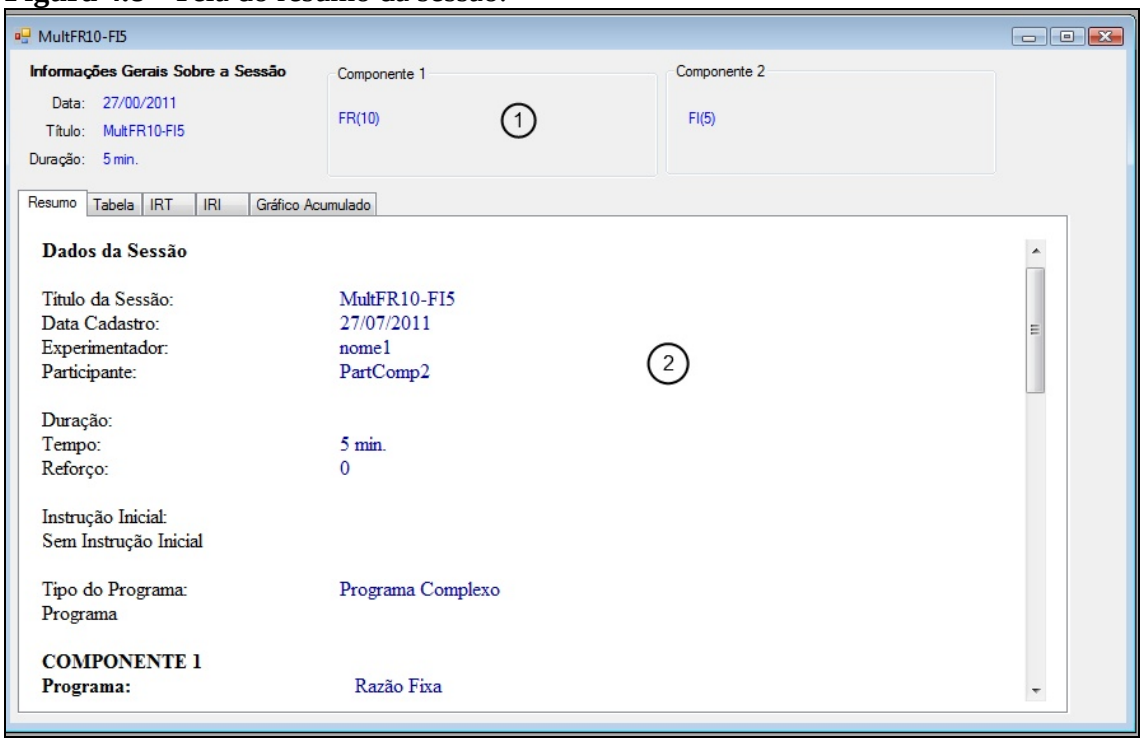


**Figura 4.9** - Página Tabela.

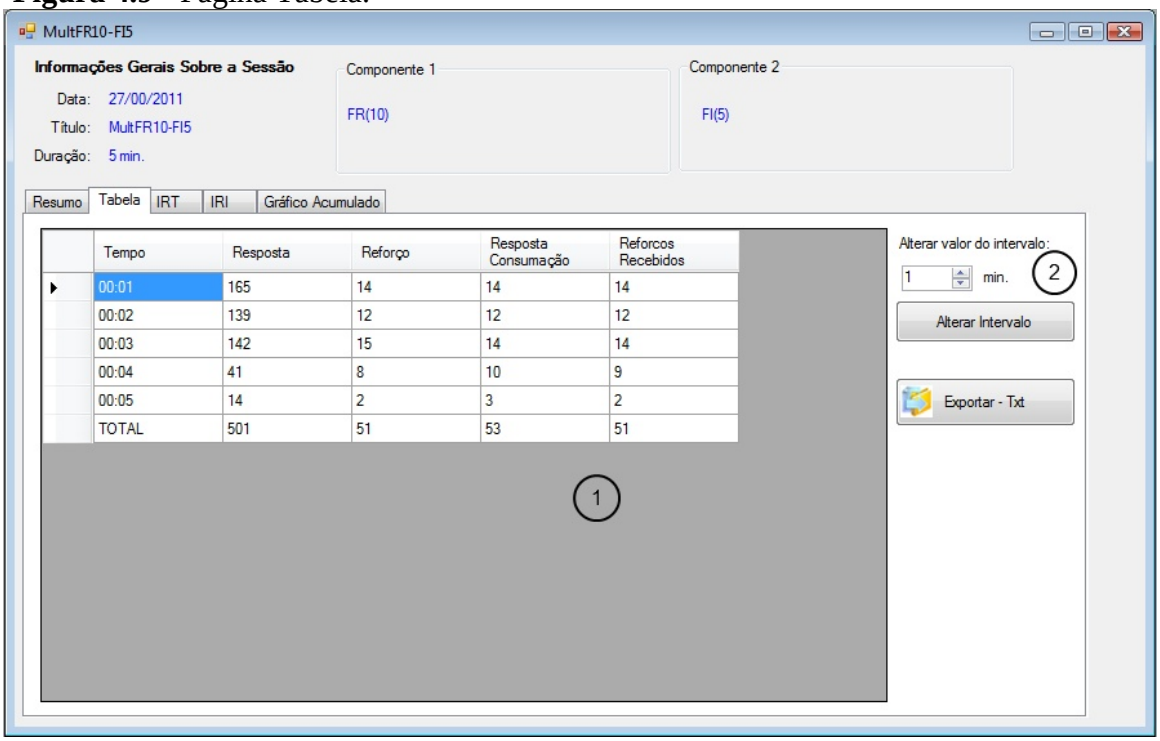



**Figura 4.10** - Página IRT

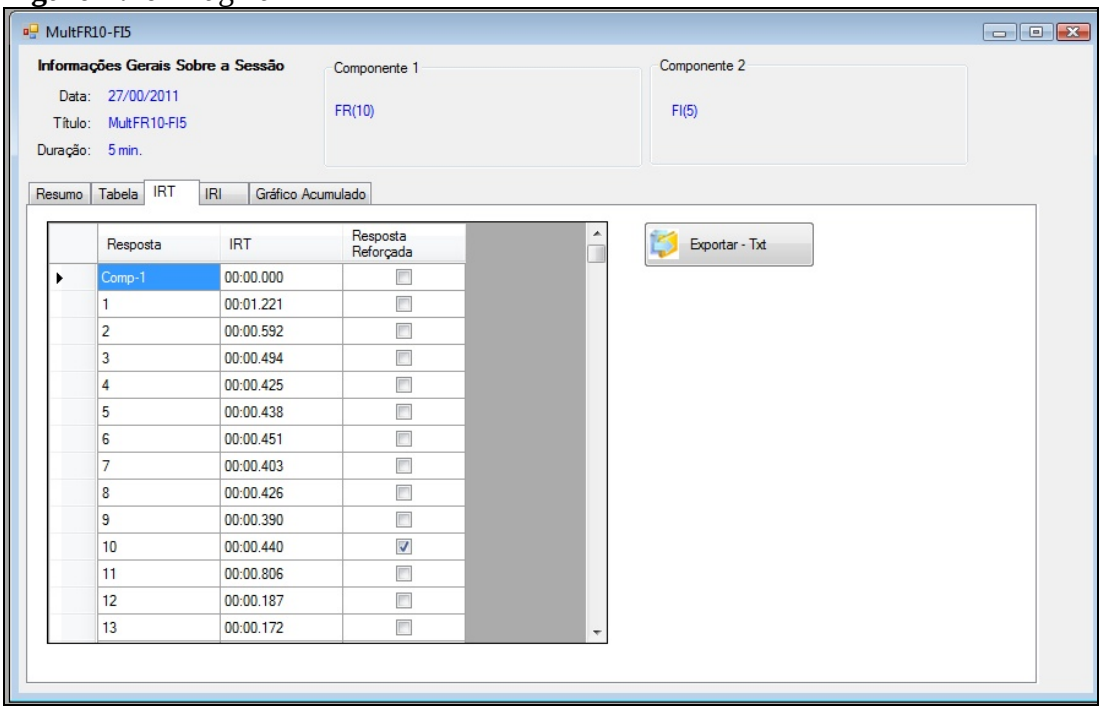
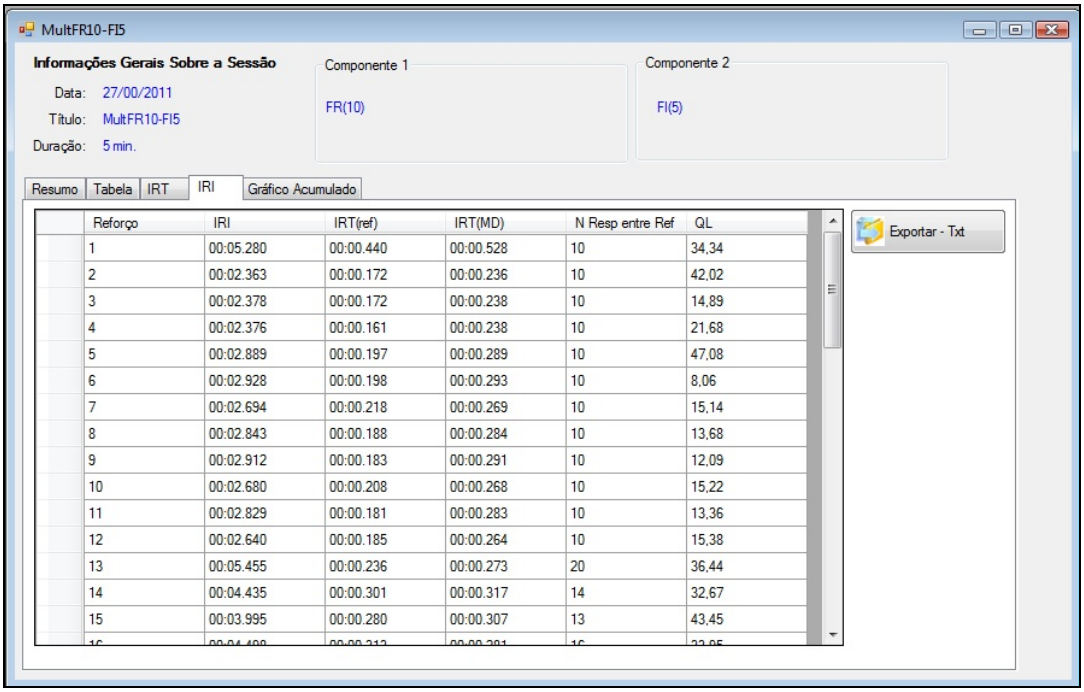


**Figura 4.11** - Página IRI




| Reforço | IRI | IRT(ref) | IRT(MD) | N Resp entre Ref | QL |
|---|---|---|---|---|---|
| 1 | 00:05.280 | 00:00.440 | 00:00.528 | 10 | 34,34 |
| 2 | 00:02.363 | 00:00.172 | 00:00.236 | 10 | 42,02 |
| 3 | 00:02.378 | 00:00.172 | 00:00.238 | 10 | 14,89 |
| 4 | 00:02.376 | 00:00.161 | 00:00.238 | 10 | 21,68 |
| 5 | 00:02.889 | 00:00.197 | 00:00.289 | 10 | 47,08 |
| 6 | 00:02.928 | 00:00.198 | 00:00.293 | 10 | 8,06 |
| 7 | 00:02.694 | 00:00.218 | 00:00.269 | 10 | 15,14 |
| 8 | 00:02.843 | 00:00.188 | 00:00.284 | 10 | 13,68 |
| 9 | 00:02.912 | 00:00.183 | 00:00.291 | 10 | 12,09 |
| 10 | 00:02.680 | 00:00.208 | 00:00.268 | 10 | 15,22 |
| 11 | 00:02.829 | 00:00.181 | 00:00.283 | 10 | 13,36 |
| 12 | 00:02.640 | 00:00.185 | 00:00.264 | 10 | 15,38 |
| 13 | 00:05.455 | 00:00.236 | 00:00.273 | 20 | 36,44 |
| 14 | 00:04.435 | 00:00.301 | 00:00.317 | 14 | 32,67 |
| 15 | 00:03.995 | 00:00.280 | 00:00.307 | 13 | 43,45 |

**Figura 4.12** - Página do Gráfico – Registro cumulativo das respostas.

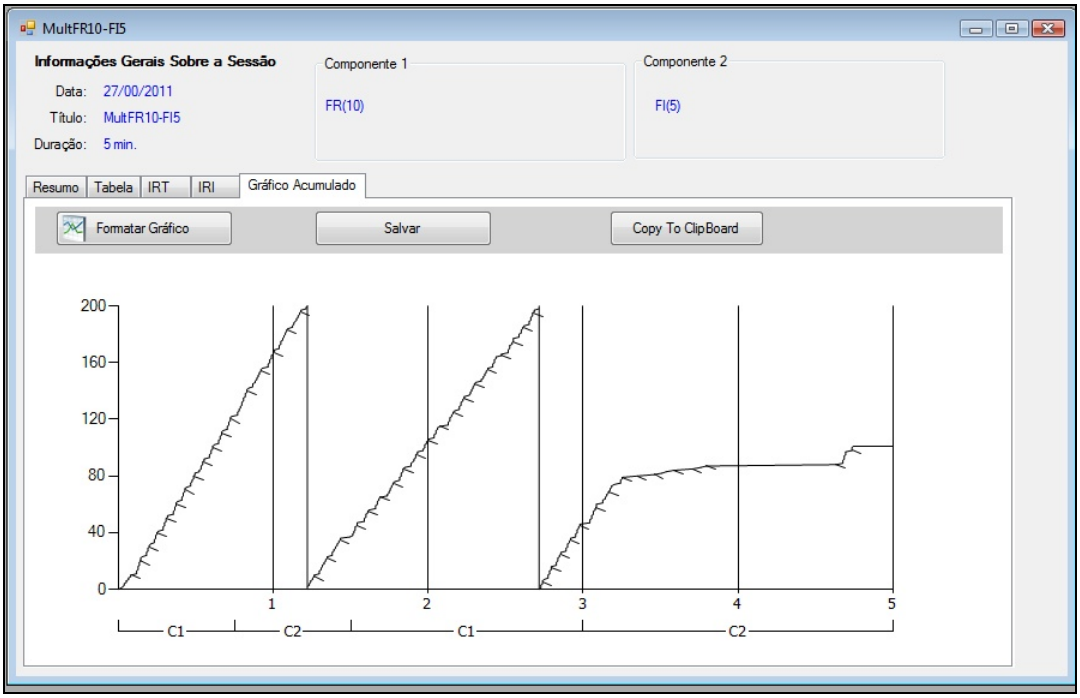


**Figura 4.13** - Formulário de configuração do gráfico.